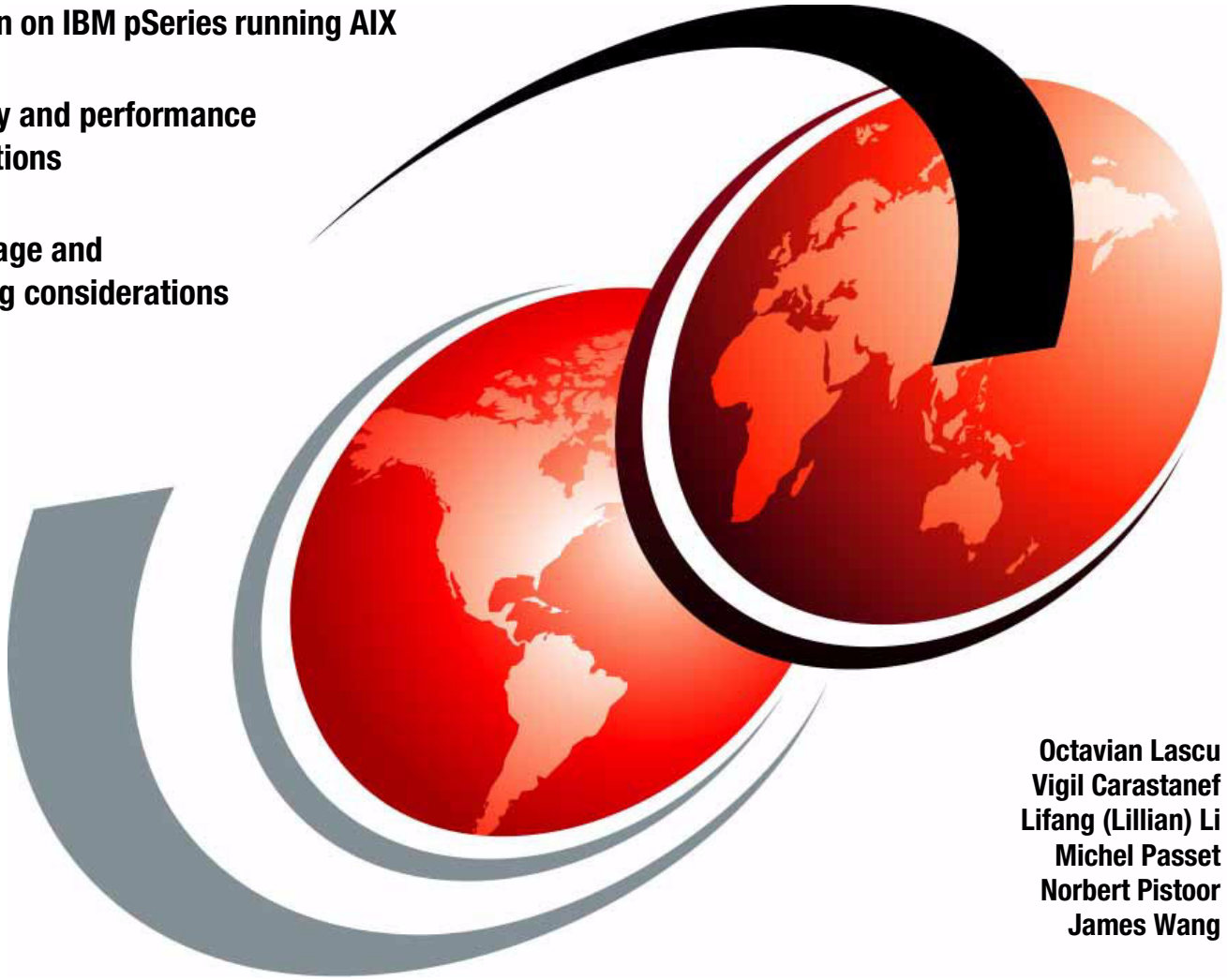
IBM

Deploying Oracle 9*i* RAC on IBM @server Cluster 1600 with GPFS

- Oracle9*i* RAC cluster planning and installation on IBM pSeries running AIX
- Availability and performance considerations
- GPFS storage and networking considerations



Octavian Lascu
Vigil Carastanef
Lifang (Lillian) Li
Michel Passet
Norbert Pistoor
James Wang

Redbooks

ibm.com/redbooks

International Technical Support Organization

Deploying Oracle 9*i* RAC on IBM @server Cluster 1600 with GPFS

October 2003

SG24-6954-00

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

First Edition (October 2003)

This edition applies to Version 5, Release 2, Modification 01 of AIX and Version 9.2.0.x of Oracle9i Real Application Clusters.

Contents

Notices

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

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

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

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

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

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

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

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

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

Trademarks

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

AIX 5L™
AIX®
DB2®
e-business on demand™
eServer™
Enterprise Storage Server®
IBM®
Micro Channel®
POWERparallel®
pSeries™
Redbooks™
Redbooks(logo) ™
RS/6000®
TotalStorage®
Versatile Storage Server™
xSeries®

The following terms are trademarks of other companies:

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

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

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

C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.

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

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

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

Preface

This IBM Redbook explores the steps for installing Oracle 9*i* RAC in an IBM @server Cluster 1600. GPFS is implemented in the Cluster as the file system of choice for Oracle code and database files.

Performance and availability considerations are presented to show system architects, system administrators, and database administrators how to implement various features of a high performance, high availability database cluster environment.

This book also presents a set of high availability tests and scenarios to help in considering these aspects in the planning and implementation phases.

The team that wrote this redbook

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

Octavian Lascu is a Project Leader at the International Technical Support Organization, Poughkeepsie Center. He writes extensively and teaches IBM classes worldwide on all areas of pSeries™ clusters and Linux. Before joining the ITSO in 2002, Octavian worked in IBM Global Services Romania as SW and HW Services Manager. He holds a Master's Degree in Electronic Engineering from Polytechnical Institute in Bucharest, and is a Certified Advanced Technical Expert in AIX/PSSP/HACMP. He has worked with IBM since 1992.

Vigil Carastanef is an IT Architect at the telecommunication company Orange SA, Romania. He has over six years of experience in the RS/6000® and pSeries fields. He works in designing and sizing architectures involving Oracle products on IBM platforms. His area of expertise includes performance tuning and high availability. He holds a degree in Electrical Engineering.

Lifang (Lillian) Li is a senior IT specialist in providing in-depth pre-sales technical support in Guangzhou, China. She has eight years of experience in application development and technical support. She holds a Master's Degree in Computer Science and was certified as IBM RS/6000 Advanced Technical Expert in 2000. She has worked at IBM for four years. Her areas of expertise include benchmarking, performance tuning, and problem determination.

Michel Passet is a benchmark manager at the EMEA PSSC center in Montpellier, France. He manages benchmarks with Oracle tuning on an AIX® pSeries environment. He has over eight years of experience with Oracle, AIX, and UNIX platforms. His areas of expertise include designing highly available infrastructures for worldwide customers. He also provides on-site technical support for migrations to AIX. He holds a degree in Computer Science Engineering. He has written numerous white papers on RAC installation on AIX.

Norbert Pistoor is an Advisory IT Specialist with eServer pSeries Presales Technical Support in Munich, Germany. He has more than 12 years of experience with RS/6000 and pSeries, including seven years in benchmarking and performance analysis. He holds a PhD in physics from the University of Mainz, Germany.

James Wang is a Senior Engineer with the IBM Design Center for e-business on demand™ in Poughkeepsie, NY. He has over 10 years of database experience with pSeries and AIX. He is an Oracle9*i*-certified DBA, an SAP-certified UNIX/Oracle Consultant, an IBM-certified

specialist in AIX and HACMP, an IBM-certified solution expert for Business Intelligence, and an IBM-certified Advanced Technical Expert on DB2® for Clusters. His areas of expertise include database system design, implementation, tuning, and high availability. He holds a PhD in Chemical Engineering and a Master's Degree in Computer Science from Louisiana State University.

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

Gordon McPheeters
IBM Poughkeepsie

Paul Moyer
IBM Poughkeepsie

Michael K Coffey
IBM Poughkeepsie

Laurent Montaron
IBM Beaverton

Rick Piasecki
IBM Austin

Ronald Bautista
IBM Oakland

Thierry Chenillot
IBM France

Christine O'Sullivan
IBM France

Dino Quintero
International Technical Support Organization, Poughkeepsie Center

Become a published author

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

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

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

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

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

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

  ibm.com/redbooks

- Send your comments in an Internet note to:

  redbook@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. JN9B Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

1

Introduction

In this chapter we present the concept of clustered servers and the benefits of a multi-server environment, such as enhanced availability and scalability—with their relevance for load sharing and parallel processing.

We also provide a short overview of the basic features of Oracle's clustered database, Oracle9*i* Real Application Clusters (RAC), explained in more detail in later chapters of this book.

The basic building blocks needed to implement a cluster running Oracle9*i* RAC are also presented. We list some key features required by hardware components such as servers, storage, and networks, and introduce some of the software products that can be used to implement an Oracle9*i* RAC in a pSeries cluster environment.

1.1 Why clusters?

Since their introduction, electronic computers have become more and more powerful with every new generation. What had been a fully equipped computing center a decade ago, can now be taken away in the size of a laptop. In certain situations, applications running on high-end servers do not generate a continuous computing workload to fully use the processing capabilities of the hardware. Therefore, server consolidation became a hot topic in many computing environments and, along with it, the idea of splitting the huge server into logically independent subunits called partitions (logical partitions).

Each logical partition (LPAR) contains processors, memory, and I/O components, and runs its own operating system. It can process a workload that could otherwise not use the unpartitioned server without wasting most of its computing power. The current versions of HW and SW also provide the ability to dynamically move resources between logical partitions without the need to stop or shut down, allowing for resource balancing and future growth, and thus offering investment protection.

While logical partitioning distributes resources between applications running in logically independent environments on the same server, clustering, on the other hand, combines several independent servers (called cluster nodes) into one logical entity, called *cluster*. Clustering is achieved by connecting nodes through one or more networks and adding a software layer that maintains and exchanges information cluster-wide. This information can be used to coordinate tasks running on different cluster nodes and distribute workload over several servers, thereby enhancing availability and improving application performance.

1.1.1 Availability

Nodes in a cluster periodically exchange information about themselves and their network connections. Status changes trigger specific cluster events that can start predefined actions. A common information base is also maintained and made available to the applications running on cluster nodes. This can be used to enhance application availability and improve continuity of operations.

For example, if a node loses connection to one of the cluster-defined networks due to a failing network adapter or cable, an event is triggered that automatically configures another adapter to replace the failed one. If no spare adapter exists, it can instead trigger a graceful stop of the application, then shut down the node, and restart the application on another cluster node. By maintaining the same network identity (IP address and MAC address), this operation can be performed transparently for any client connected to the application through the affected network.

Another way to enhance application availability is by running a copy of the application on each cluster node, and distributing the client connections among them. Should one of the nodes fail, the clients affected could reconnect to one of the surviving nodes. This approach is often found in three-tier applications (backend—database, middleware—application servers, and clients) specially for the application server.

One common example is a multi-node cluster Web server where client connections are managed by a network dispatcher in round-robin fashion.

1.1.2 Scalability

A clustering environment also provides for future growth. You can start by running an application on a standalone machine, sized for a specific workload, and, if the application

needs additional resources, additional hardware (servers, storage, and networks) can be added to support the increased workload.

To use this approach, the application design must be able to use distributed resources (among several servers). Many algorithms, however, are optimized for running on a standalone server. Splitting the application into several pieces in order to run them in parallel on multiple servers may require a complete application redesign. For this reason, special editions of the application software are available for clustering environments.

1.1.3 Load sharing

Once the problem of parallelizing the application has been solved, the enhanced availability and scalability of the new application design can be exploited by implementing a load sharing mechanism for the clustered application.

Clients connecting to the application are directed to a node in the cluster according to a predefined algorithm. Load sharing algorithms can use a round-robin scheme (where the cluster nodes are always selected according to a predefined sequence), or use more sophisticated, feedback-based mechanisms (where the server nodes are selected based on resources availability and prior load characterization). Properly implemented, this results in using the nodes more evenly, and thereby optimizing overall application performance.

Load sharing also provides for enhanced availability. Should one cluster node fail, the load sharing algorithm would then only consider the remaining nodes for initiating new connections and distribute the clients among them. The existing clients would encounter an error and could react to this by reconnecting to one of the available nodes, repeating any unfinished transactions.

This applies to an environment where a large number of transactions is performed concurrently on the same set of data. Although each individual transaction can be performed on a standalone node, the number of concurrent transactions (and the need for reasonable response time) may require a cluster application implementation.

1.1.4 Parallel processing

Certain types of transactions may exhaust the resources of a single server and have very long turnaround times, even if the application that processes those transactions runs alone on that server. This can be found, for example, in scientific and technical environments, or in data warehousing applications, where large amounts of data have to be scanned and processed.

If it is possible to split the task into several subtasks that can be performed independently of each other (for example, if they operate on distinct subsets of data), a cluster of servers can be used to parallelize the transaction (at least partially) by executing the independent subtasks on different cluster nodes, in parallel. This can result in significantly reducing the transaction response time. Depending on application-specific internals, a different algorithm may have to be implemented to exploit the parallel design.

For this type of application, the cluster software usually provides the framework that is used by the application to coordinate the different subtasks, establish communication channels at the application level, and collect the final results.

1.2 Oracle9*i* RAC overview

Oracle9*i* Real Application Clusters (RAC) is the parallel version of the Oracle9*i* database software. It makes it possible to run multiple instances (each on a separate cluster node) for accessing the same database.

Oracle9*i* RAC achieves this task by using some features of the underlying cluster software and operating systems.

Since the different instances have to access the same database files, a shared storage space is required (which can be accessed from all cluster nodes, concurrently). This can be provided by hardware (for example, via fiber-attached storage), or by an additional software layer which provides shared storage access via network to other nodes in the same cluster (for example, virtual shared disks - VSD).

The shared storage space can be used as raw devices, or by implementing a cluster file system. The latter approach tends to simplify certain maintenance tasks and provides a similar “look and feel” as a “traditional” non-clustered database on a standard file system.

In both approaches, concurrent access to the same data by different nodes is required and must be implemented using proper locking mechanisms. Oracle9*i* RAC provides its own locking infrastructure and does not rely on the cluster software, file system, or the operating system, for handling the locks.

Database instances running on different cluster nodes require a fast (high bandwidth, low latency) and reliable (highly available) interconnect network for transferring locking information and exchanging database blocks.

In the early days of Oracle Parallel Server (Oracle9*i* RAC predecessor), database instances running on cluster nodes could not exchange data blocks directly between their memory caches. Data blocks from one instance had to be written to a shared disk zone by the first instance and then read by the other instance.

This mechanism has changed in Oracle9*i* RAC. The concept of “Cache Fusion” has been introduced, which allows any instance to access the data blocks in other instances’ caches by transferring them via the fast interconnect.

Multiple instances of the same database provide application redundancy. Clients can connect to any available instance and submit their transactions. This enhances the availability of the application in case of a node failure, and allows load balancing between the instances.

Oracle9*i* RAC supports running applications designed for single instance (standalone) databases. Although an application will work without modifications and will produce correct results, it may not offer optimal performance. In some cases (for example, when the same data block is updated frequently by different instances), running such an application in a cluster can be slower than running it in a standalone database. This is due to the overhead incurred by the coordination mechanism for certain types of transactions.

References

Oracle9i Real Application Clusters: Concepts, Release 2 (9.2), March 2002

Oracle9i Real Application Clusters: Deployment and Performance, Release 2 (9.2), March 2002

1.3 Cluster building blocks

In this section we describe some basic building blocks that can be used to create an IBM @server pSeries cluster suitable for use with Oracle 9i RAC. Building blocks considered are hardware as well as software components used in IBM pSeries cluster environments.

1.3.1 Hardware components

The basic hardware components are servers, storage, and networks.

Servers

Cluster nodes can be medium-to-high end SMP servers (such as IBM eServer p690, p670, p655, and so on), or logical partitions (LPARs) inside the same type of servers.

A minimum of two nodes is required, and, to benefit from the enhanced availability and scalability of the cluster, it is recommended that the nodes should be configured symmetrically (that is, each should include the same number of processors, memory, and optional features). For reasons of availability, we recommend not to have only one physical server with two LPARs, although this might be used for testing or development purposes.

Storage

Storage subsystems that can be attached to more than one server at a time include IBM 7133 SSA disks as well as Fibre Channel (FC) attached storage such as IBM ESS or IBM FAStT models.

Using SSA adapters with the Fast Write Cache option, up to two nodes can be directly attached to the same set of disks. For larger clusters, fiber channel storage is the only supported direct attached shared disk configuration.

Alternatively, concurrent access to the disk storage from more than two nodes can be provided by additional software, such as IBM Virtual Shared Disk.

Networks

The clustering software (such as IBM HACMP for AIX) requires several networks connecting the cluster nodes. All networks known to the cluster are used to exchange status information and, by doing so, to verify proper network and node functionality.

One of these networks should be a non-IP network in order to ensure that communication is still possible even if the TCP/IP subsystem is no longer operational. Examples of non-IP networks are RS232 serial line, target mode SSA, target mode SCSI, and the new disk heartbeat network, available in HACMP 5.1.

For Oracle9*i* RAC, a fast interconnect network is required, which should be a high speed, low latency, switched network. Examples of interconnect are Gbit Ethernet, SP Switch, SP Switch2, etc. More than one network can be configured to provide redundancy.

The client network must provide enough bandwidth to allow clients to connect and exchange data with the server with a reasonable response time. Aspects of availability and scalability have to be addressed here as well.

For a typical environment, we recommend a switched network with redundant components, to provide the necessary failover functionality in case one of the network components fails or needs maintenance.

1.3.2 Software components

This section gives an overview of the main software components for building the cluster for Oracle 9i RAC.

PSSP (Parallel Systems Support Program)

PSSP is a collection of applications to manage an IBM pSeries or RS/6000 SP cluster as a parallel processing system. It provides a single point of control for administrative tasks. It is used for SP systems, but can also be used to build an IBM Cluster 1600 comprising of pSeries servers or a mixture of pSeries servers and SP nodes.

VSD (Virtual Shared Disk)

VSD currently is a subsystem of PSSP. It can be used in IBM Cluster 1600 environments managed by PSSP using a fast interconnect to transfer data between the nodes.

VSD provides all nodes in the cluster with concurrent access to data stored on disks that may not be physically attached to the local node. VSD nodes are servers (which have direct attached storage) and clients (which access the storage on the server nodes via a network).

RSCT (Reliable Scalable Cluster Technology)

RSCT is part of the AIX 5L™ operating system. It includes topology and group services and provides the high availability infrastructure (cluster membership and event generation) used to combine several nodes into a cluster. This is the basic cluster configuration, also known as an RSCT peer domain (RPD) cluster.

HACMP/ES (High Availability Cluster MultiProcessing/Enhanced Scalability)

HACMP/ES for AIX is the IBM tool for building AIX-based, mission-critical computing platforms. It can be used with most IBM pSeries models as well as with SP nodes.

For Oracle9*i* RAC, HACMP/ES for AIX is required to provide cluster definition and event management functionality. If the database tablespaces are stored on raw devices, the Concurrent Resource Manager (CRM) component of HACMP/ES for AIX is also required.

GPFS (General Parallel File System)

GPFS for AIX 5L is the shared file system running in a pSeries or RS/6000 SP cluster environment. GPFS can be implemented in:

- A PSSP cluster using VSDs for shared disk access
- An RSCT peer domain with direct attached storage (to all nodes in the cluster)
- An HACMP/ES cluster with direct attached storage (to all nodes in the cluster)

With Oracle9*i* RAC, GPFS can be used to store the database files (instead of RAW devices) as well as the database executables and configuration files (one common repository for all nodes instead of separate copies for each node).

2

Typical Oracle9*i* RAC configurations

This chapter describes some typical configurations for running Oracle9*i* RAC on IBM @server pSeries clusters. Oracle9*i* RAC relies on a shared storage architecture. All nodes in the cluster access the same physical database. We describe the three typical configurations used for implementing Oracle9*i* RAC on IBM pSeries or RS/6000 platforms:

- Storage on RAW devices
- Storage on Virtual Shared Disk (VSD)
- Storage on General Parallel File System (GPFS)

2.1 Basic building blocks

All configurations are based on the following building blocks:

- Hardware
  - Server nodes
  - Storage
  - Networking
- Software
  - Operating system
  - Cluster software
  - Oracle9*i* RAC (application)

2.2 Oracle9*i* RAC on RAW devices

This configuration is suitable for those who prefer to store the Oracle database files on RAW devices. It offers good performance, since direct access to disk is provided, but is fairly complicated to administer. In this configuration, all cluster nodes are directly connected to the storage subsystem.

2.2.1 Hardware requirements

Server nodes

- IBM pSeries servers (full SMP or LPAR), SP nodes, legacy RS/6000 servers

  The restriction is that Oracle9*i* RAC is a 64-bit SW that requires 64-bit HW to run.

> **Note:** When running Oracle9*i* RAC on LPAR nodes, it is recommended to have LPAR cluster nodes located on separate pSeries servers in order to avoid single points of failure, such as power supply, Central Electronic Complex (CEC), system backplane, etc.

Storage systems

Storage subsystems (which must support concurrent access), such as:

- IBM 7133 Serial Storage Architecture (SSA) Disk Subsystem

  This is the IBM “legacy” serial storage architecture. It has built-in redundancy at adapter and communication path level and provides good performance. Its main limitation is the number of systems capable of concurrently accessing the SSA storage.

  The number of adapters that are supported in an SSA shared disk configuration is shown in Table 2-1.

- IBM Enterprise Storage Server® (ESS)
- IBM TotalStorage® FAStT200, FAStT500, FAStT700 and FAStT900

Table 2-1 SSA configuration

Array type	Number of adapters in loop	Type of adapters supported
Non-RAID	8	Advanced SerialRAID Adapter
		PCI SSA Multi-Initiator/RAID EL Adapter
		Micro Channel® SSA Multi-Initiator/RAID EL Adapter
RAID-0	1	Advanced SerialRAID Adapter
RAID-1	2	Advanced SerialRAID Adapter at microcode level above 5000
RAID-5	2	Advanced SerialRAID Adapter
		PCI SSA Multi-Initiator/RAID EL Adapter
		Micro Channel SSA Multi-Initiator/RAID EL Adapter
RAID-10	2	Advanced SerialRAID Adapter at microcode level above 5000
Fast-Write	2	Advanced SerialRAID Adapter at microcode level above 5000

For detailed information on Advanced SerialRAID Adapter, refer to:

Advanced SerialRAID Adapters User's Guide and Maintenance Information, SA33-3285

Also, for information about SSA Disk Subsystem, see:

`http://www.storage.ibm.com/hardsoft/products/ssa/docs/index.html#rs6k`

For the latest FAStT Storage interoperability matrix, see:

`http://www.storage.ibm.com/disk/fastt/supserver.htm`

Network

- IP Networks
  - Oracle interconnect: Gigabit Ethernet, 10/100Mbit Ethernet, SP Switch, SP Switch2

    Although 100 Mbit Ethernet can be used for Oracle interconnect, we highly recommend at least one Gigabit Ethernet network.

    Since communication between Oracle instances uses the Oracle interconnect network, its bandwidth decides the performance of Oracle9*i* RAC.

    Also, for high availability purposes, we recommend to define two networks for Oracle interconnect traffic, thus ensuring network failover in case one network fails.
  - Client access network: standard 10/100Mbit Ethernet
- Serial (non-IP)
  - HACMP non-IP heartbeat network

    It is strongly recommended that a non-IP network be present between cluster nodes to eliminate TCP/IP subsystem failure ("split brain"). The supported non-IP network types are RS-232, target mode SCSI (tmSCSI), or target mode SSA (tmSSA).

2.2.2 Software requirements

Operating system

- AIX 5.2 or AIX 5.1 or AIX 4.3.3;

  The AIX operating system provides the infrastructure for the upper software layers (TCP/IP, storage drivers etc.).

Cluster software

- HACMP/ESCRM 4.5 or HACMP/ESCRM 4.4.1

  HACMP/ESCRM provides applications with high availability services and concurrent storage (shared storage access) on IBM eServer pSeries platforms.

  HACMP/ESCRM is based on the RSCT layer (included in base OS) and provides the "glue" for managing an HACMP cluster environment, such as:

  - Cluster membership
  - Configuration and cluster nodes management
  - Failover or fallback behavior of resource groups
  - Application monitoring

  For further information about HACMP/ESCRM, refer to the product documentation:

  – *High Availability Cluster Multi-Processing for AIX Planning Guide Version 4.5,* SC23-4277
  – *High Availability Cluster Multi-Processing for AIX Enhanced Scalability Installation and Administration Guide Version 4.5,* SC23-4306
  – *High Availability Cluster Multi-Processing for AIX Enhanced Scalability Installation and Administration Guide Version 4.4.1,* SC23-4279

  See Table 2-2 for the current certification matrix (at the date this book was written).

Table 2-2 Oracle9i RAC certification matrix with HACMP/ESCRM

Oracle product version	AIX version	HACMP/ESCRM version
Oracle9*i* RAC 9.0.1 64-bit	4.3.3	4.4.x
Oracle9*i* RAC 9.2 64-bit	4.3.3	4.4.x
	5.1	4.4.x
	5.1	4.5
	5.2	4.5

For the latest certification matrix, see the following URL:

`http://otn.oracle.com/support/metalink/content.html`

2.2.3 Application architecture

Oracle9*i* RAC on RAW devices is based on a shared disk architecture. Figure 2-1 on page 11 shows a two-node cluster. The lower solid line is the primary Oracle interconnect, the middle dashed line is the secondary Oracle interconnect. For high availability, both these networks should be defined in the HACMP as "private".

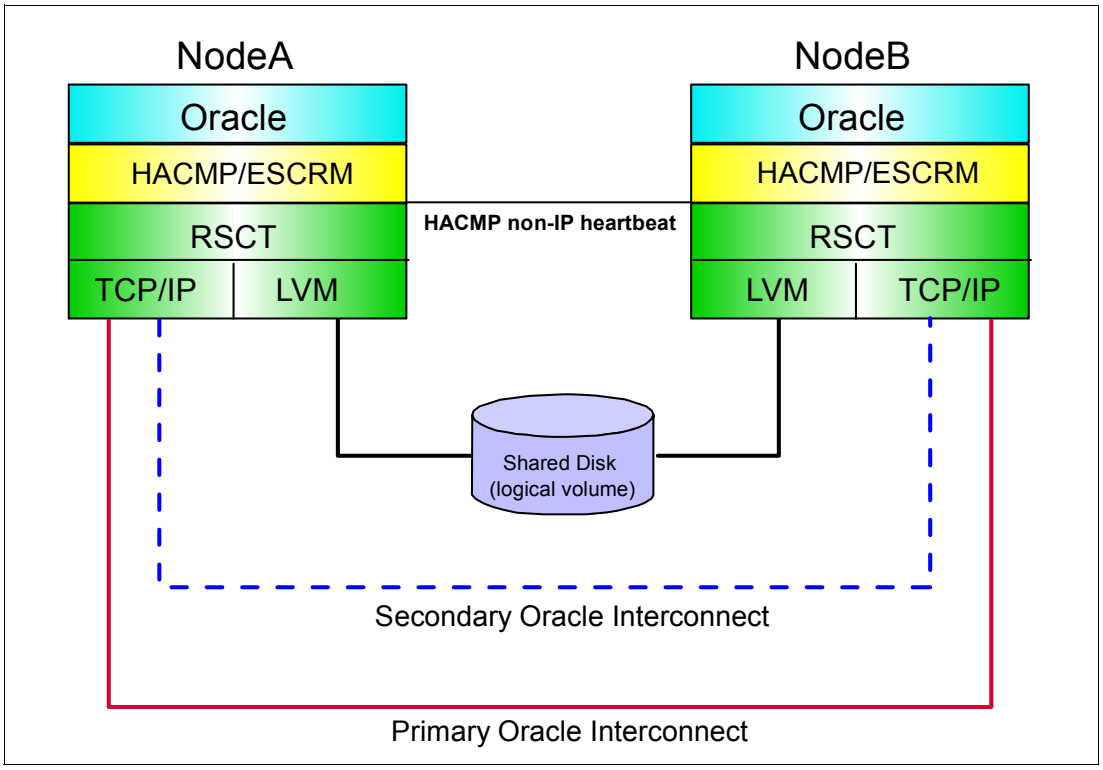


Figure 2-1 HACMP and RAW devices configuration

HACMP/ESCRM provides Oracle9*i* RAC with the infrastructure for concurrent access to disks. Although HACMP provides concurrent access and a disk locking mechanism, this mechanism is not used. Oracle, instead, provides its own locking mechanism for concurrent data access, integrity, and consistency.

Volume groups are varied on all the nodes, thus ensuring short failover time. This type of concurrent access can only be provided for RAW logical volumes (devices). HACMP/ESCRM does not support concurrent file systems.

Oracle datafiles use the RAW devices located on the shared disk subsystem. In this configuration, an HACMP resource group has to be defined to handle the concurrent volume groups.

HACMP can also provide additional functionality, such as:

- An automatic mechanism for application start and stop (Oracle instance)
- Custom application monitoring, which can be used to monitor the Oracle instance
- Event notifications

Since Oracle requires HACMP CLVM (for activating volume groups in concurrent mode), the number of cluster nodes for this configuration (depending on the storage used) is limited to 16.

Furthermore, for the SSA Disk Subsystem, the maximum number of nodes is 8 (see Table 2-1 on page 9).

In conclusion, this configuration (RAW devices) is suitable for running Oracle9*i* RAC, because it provides the fastest disk access, but is limited to 16 nodes (8 with SSA).

Another drawback of this configuration stems from the fairly complex administrative tasks, such as maintaining datafiles, Oracle code, and backup/restore operations.

Note: In this solution, neither GPFS nor RPD clusters are used.

Figure 2-2 shows the software layers involved in this configuration. The Subsystem Device Driver (SDD) layer is not mandatory, but can provide additional performance and availability if used in conjunction with redundant SAN architecture.

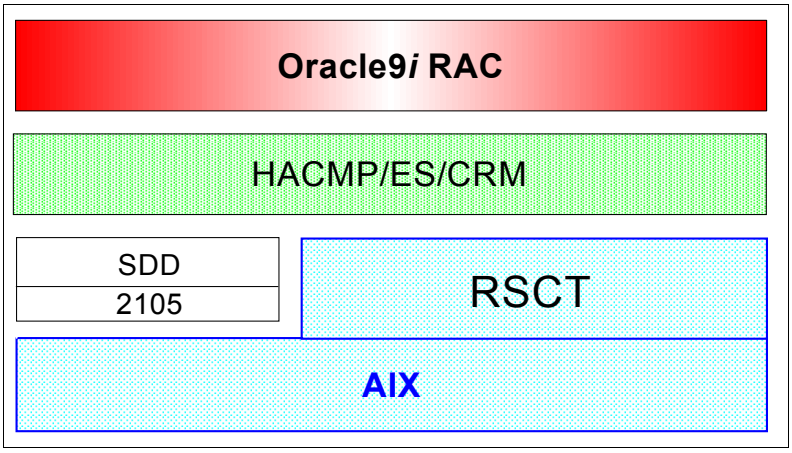


Figure 2-2 RAW devices with HACMP/ESCRM software architecture

2.3 Oracle9*i* RAC on VSD

This configuration is suitable in a high performance computing (HPC) environment, where scalability is a key factor and performance is critical. It is fairly complex and requires special hardware and software.

This configuration is based on the same building blocks listed in 2.1, “Basic building blocks” on page 8.

2.3.1 Hardware requirements

Server nodes

IBM pSeries servers (full SMP or LPAR), SP nodes, 64-bit legacy RS/6000 servers (supported by PSSP - Parallel System Support Programs for AIX).

Refer to the following documents about the hardware requirements for an SP system:

- *RS/6000 SP Planning Vol. 1, Hardware and Physical Environment,* GA22-7280
- *RS/6000 SP Planning, Vol. 2, Control Workstation and Software Environment,* GA22-7281

Also see the following URL about hardware planning for an SP system:

```
http://www-1.ibm.com/servers/eserver/pseries/library/sp_books/planning.html
```

Storage systems

Storage systems that support VSD are:

- IBM 7133 SSA Disk Subsystem
- IBM Enterprise Storage Server (ESS)

- IBM 2105 Versatile Storage Server™ (VSS)

See the following URL for the latest hardware information:

`http://www-1.ibm.com/servers/eserver/pseries/library/sp_books/hardware.html`

Network

- Oracle interconnect: Gigabit Ethernet, SP Switch, SP Switch2, 10/100Mbit Ethernet

  SP Switch provides a high-speed, low-latency message-passing network that connects processor nodes.

  SP Switch2 is a next-generation switch for an SP system. It provides a low-latency, high-bandwidth, message-passing network that interconnects the nodes.

  Although 10/100 Mbit Ethernet can be used for Oracle Interconnect, we highly recommend that you use at least one Gigabit Ethernet network if you don't have the switch.

  Also, for high availability purposes, we recommend that you define two networks for Oracle interconnect traffic, thus ensuring network failover in case one network fails.
- VSD network: Gigabit Ethernet, SP Switch, SP Switch2, 10/100Mbit Ethernet
- Client access network: standard 10/100 Mbit Ethernet

2.3.2 Software requirements

Operating system

- AIX 5.1 or AIX 4.3.3

Cluster software

- PSSP 3.5, or PSSP 3.4, or PSSP 3.2

  PSSP is a comprehensive suite of applications that runs on SP nodes. It provides a single point of control for the administrator to view, monitor, and operate the system.

  For more detail about PSSP, refer to:

  `http://www.ibm.com/servers/eserver/pseries/library/sp_books/pssp.html`

  The Oracle9*i* RAC certification matrix with PSSP now is as shown in Table 2-3.

Table 2-3 Oracle9i RAC certification matrix with PSSP

Oracle product version	AIX version	PSSP version
Oracle9*i* RAC 9.0.1 64-bit	4.3.3	3.2
		3.4
Oracle9*i* RAC 9.2 64-bit	5.1	3.4
		3.5(projected)

Note: If SP switch is used, PSSP version 3.4 is required.

For the latest certification matrix, see:

`http://otn.oracle.com/support/metalink/content.html`

2.3.3 Application architecture

The Oracle9*i* RAC on VSD solution is also a shared disk architecture (provided by VSDor one of its derivates - RVSD, HSD). Figure 2-3 on page 14 shows this architecture with a three-node cluster. The solid line is the primary Oracle interconnect network (SPS, SPS2 od Gbit Ethernet), the coarse dashed line is the secondary Oracle interconnect (Gbit Ethernet, for high availability), and the green dotted line is the VSD network (in case of using SPS or SPS2, the VSD network is the same as the Oracle primary interconnect).

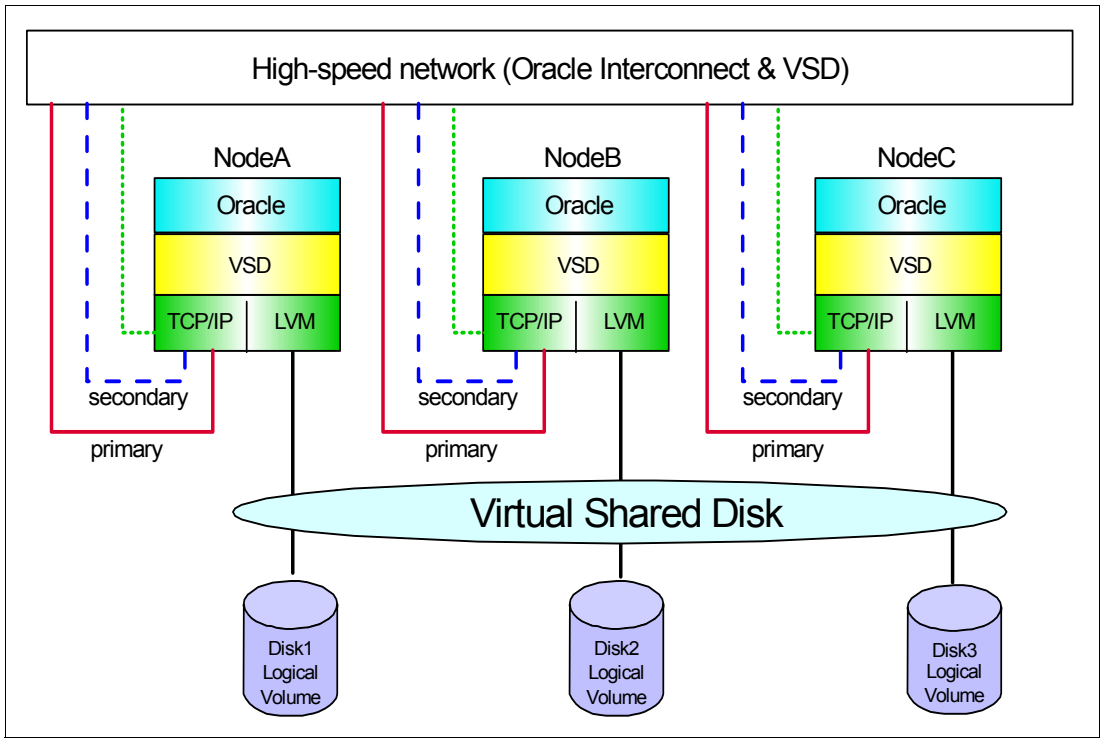


Figure 2-3 PSSP and VSD configuration

In this configuration, disks do not have to be physically attached to all nodes. Storage traffic uses the SP Switch, SP Switch2, Gigabit Ethernet, or 10/100 Mbit Ethernet.

The VSD software layer allows each node access to a logical volume as if it was configured locally. A node connected directly to the storage (which has a local logical volume corresponding to the VSD) is called a *server* node. A node that accesses other nodes' VSDs is called a *client* node. A node can be both a client node and server node. For detailed information about VSD, see:

- *Parallel System Support Programs for AIX:Managing Shared Disks Version 3 Release 5,* SA22-7349
- *RS/6000 SP Planning, Vol. 2, Control Workstation and Software Environment,* GA22-7281

If the VSD logical volume is attached locally (resides on a local node), VSD uses LVM for accessing data. If the disk is attached remotely, VSD accesses data through an IP network. VSD logical volumes are used to store Oracle datafiles, using RAW access, as in the first configuration. Of course, the Oracle home directory (which includes the Oracle software binaries), must reside on a JFS file system on disks locally attached to each node.

Although it provides the benefit of offering network shared RAW logical volumes to nodes in the cluster, this configuration does not provide highly available storage. As can be seen in

Figure 2-3, if the server node NodeA fails (which is the server for Disk1), NodeB cannot access Disk1.

For high availability VSD, we recommend that you use Recoverable Virtual Shared Disk (RVSD) and twin-tailed disks or disk arrays. A twin-tailed disk is a disk or group of disks that is attached to two nodes of a PSSP cluster.

Figure 2-4 shows this configuration. When Disk1's server node, NodeA, fails, NodeB takes over Disk1, and the client node can still access Disk1 through NodeB.

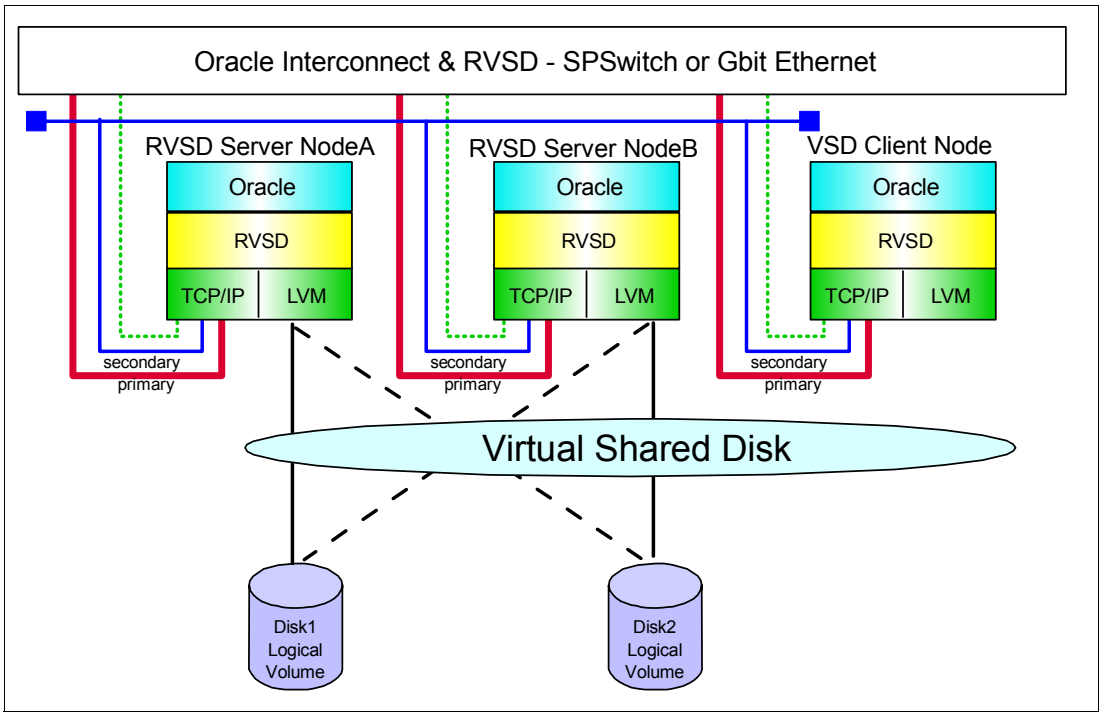


Figure 2-4 PSSP and RVSD configuration

The maximum number of nodes in this configuration is 128. This configuration is suitable when horizontal scalability is the most important factor.

Note: By default, if PSSP is installed, Oracle will use the PSSP-defined cluster. If HACMP/ESCRM is installed in the same environment and Concurrent Logical Volume Manager (CLVM) has to be used instead VSD, you must set the PGSD_SUBSYS=grpsvcs environment variable.

2.4 Oracle9*i* RAC on GPFS

This configuration is suitable for those who require simplified administration. Both Oracle software code and datafiles can be put on file systems.

Same building blocks apply to this configuration (see 2.1, "Basic building blocks" on page 8).

2.4.1 Hardware requirements

Server nodes

- IBM pSeries servers (full SMP or LPAR), SP nodes, 64-bit legacy RS/6000 servers

Restrictions apply for the hardware configuration, depending on the number of adapters to be used and the type of storage.

> **Note:** When running Oracle9*i* RAC on LPAR nodes, it is recommended to have LPARs located on separate pSeries servers in order to avoid single points of failure such as the power supply, Central Electronic Complex (CEC), system backplane, etc.

Storage systems

- SSA disks: minimum of two nodes

  Maximum of eight nodes is allowed if you use SSA Disk Subsystem as storage. For detailed information, refer to Table 2-1 on page 9.

- Fibre Channel disks

  A minimum of two nodes if ESS storage subsystem is used, or three nodes for other supported Fibre Channel (FC) attached storage.

  FC attached storage subsystems suitable for this configuration are:

  – IBM Enterprise Storage Server (ESS)
  – IBM TotalStorage FAStT200, FAStT500, FAStT700 and FAStT900

Network

- IP networks
  – Oracle interconnect: Gigabit Ethernet, 10/100M Ethernet

    Although 10/100 Mbit Ethernet can be used for Oracle Interconnect, we highly recommend that you use at least one Gigabit Ethernet.

    Also, for high availability purposes, we recommend that you define two networks for Oracle Interconnect traffic, thus ensuring network failover in case one network fails.

  – GPFS network:100 Mbit Ethernet, Gigabit Ethernet (minimum of 100 Mb per second)

    GPFS network should be a dedicated network that is not used for any other data traffic. Do not use the same network as the Oracle Interconnect network.

- Serial (non-IP)

  HACMP non-IP heartbeat network: RS-232, tmSSA, tmSCSI

2.4.2 Software requirements

Operating system

- AIX 5.2 or AIX 5.1 or AIX 4.3.3

Cluster software

- GPFS 2.1 or GPFS 1.5

  The IBM General Parallel File System (GPFS) provides users with a shared file system across all the nodes in the cluster. The operation on GPFS is the same as JFS. For detailed information, see the following GPFS documentation:

  – *General Parallel File System for AIX 5L:AIX Clusters Concepts,Planning,and Installation Guide,* GA22-7895
  – *General Parallel File System for AIX 5L:AIX Clusters Administration and Programming Reference,* SA22-7896
  – *IBM Cluster 1600 Managed by PSSP 3.5:What's New, SG24-6617*

Note: If you install Oracle9*i* RAC on GPFS 1.5 and AIX 5L, you must use AIX 32-bit kernel. If you install Oracle9*i* RAC on GPFS 2.1, you may use both 32-bit and 64-bit kernels.

- HACMP/ES 4.5 or HACMP/ES 4.4.1

  HACMP is required by Oracle for detecting node failure, adapter failure, or network failure. GPFS does not require HACMP.

  The Oracle9*i* RAC actual support matrix is shown in Table 2-4.

Table 2-4 Oracle9i RAC support matrix with GPFS & HACMP/ES

Oracle product version	AIX level	GPFS version	HACMP/ES version
Oracle9*i* RAC 9.2 64-bit	4.3.3 ML09	1.5 with APAR IY34917	4.4.1
	5.1 ML01 with APAR IY28111	1.5 with APAR IY34917	4.4.1
		2.1 with PTF U486402	4.5
	5.2 ML01	2.1 with PTF U486402	4.5

For the latest support matrix, check also:

`http://otn.oracle.com/support/metalink/content.html`

2.4.3 Application architecture

The Oracle9*i* RAC on GPFS solution is a shared file system architecture. Figure 2-5 on page 18 shows the architecture of Oracle9*i* RAC on GPFS. In this configuration, GPFS provides Oracle with a shared file system, and HACMP provides the cluster membership information for Oracle. The lower solid line is the primary Oracle Interconnect, the blue dashed line is the secondary Oracle Interconnect for high availability, and the green dotted line is GPFS network.

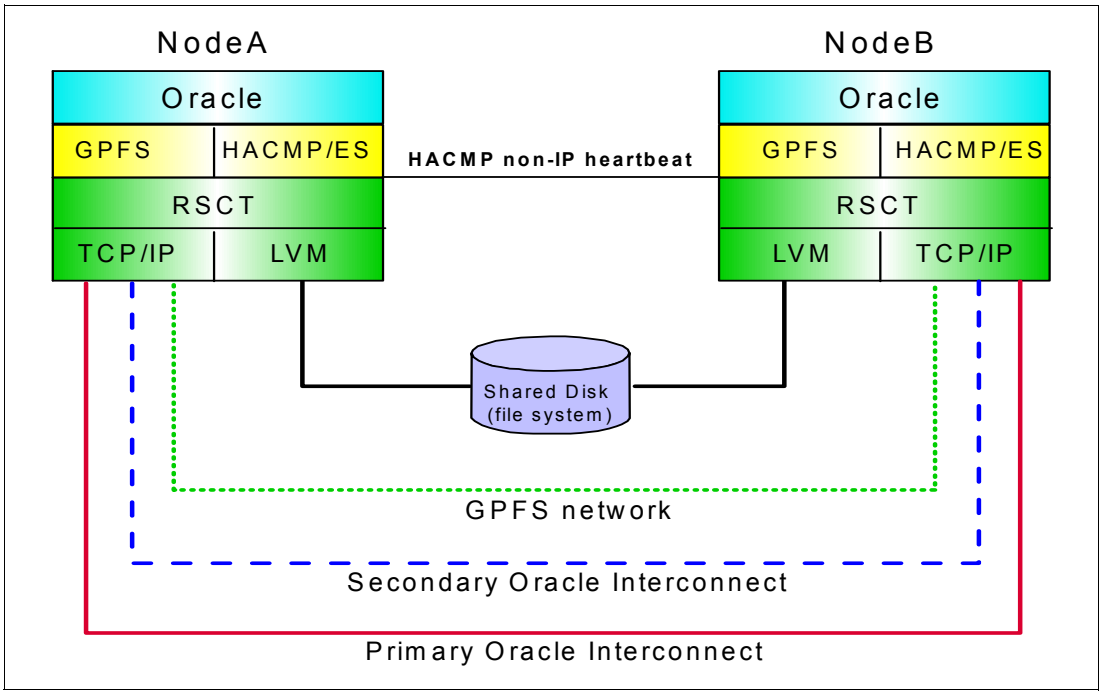


Figure 2-5 GPFS configuration

Selection of a GPFS cluster

There are three types of GPFS clusters:

- GPFS on VSD
- GPFS on RSCT peer domain (RPD) - only for GPFS 2.1

  The GPFS cluster on RPD requires Reliable Scalable Cluster Technology(RSCT). RSCT is a set of software components that provide a comprehensive clustering environment for AIX and Linux. RSCT is part of AIX 5L. The set of nodes configured for GPFS usage is called an RSCT *peer domain* (RPD) cluster.

  The RPD cluster is maintained with the RSCT commands. Refer to the following publications for more detail about RSCT:

  - *IBM Reliable Scalable Cluster Technology for AIX 5L:Administration Guide,* SA22-7889
  - *IBM Reliable Scalable Cluster Technology for AIX 5L:Technical Reference,* SA22-7890
  - *A Practical Guide for Resource Monitoring and Control (RMC),* SG24-6615
- GPFS on HACMP

  It is also possible to use the HACMP cluster definition to configure GPFS. HACMP/ES is also based on an RPD, so GPFS does not require a separate cluster definition using RSCT commands. The nodes in the GPFS cluster may be a subset of the nodes in the HACMP cluster.

Since Oracle9*i* RAC requires the HACMP/ES event management subsystem to detect node failures, adapter failures, and network failures, you can set up a GPFS cluster using the HACMP cluster definition, thus simplifying the GPFS configuration and administration tasks.

It is also possible to set up GPFS on top of an RPD cluster. This allows complete separation of the cluster file system, which may be useful for testing standalone Oracle instances regardless of the status of the HACMP subsystem. Refer to Chapter 5, “Implementing RAC over GPFS” on page 145for more detail.

Advantages of running Oracle9i RAC on GPFS

- Simplified installation

  GPFS is a shared file system, so we can keep a single image of Oracle binary files. We may put both Oracle datafiles and the Oracle Home directory, which includes the Oracle software binary files, on GPFS. This greatly simplifies the installation and administration of Oracle9*i* RAC.

- Simplified backup and recovery

  Since all the datafiles can be stored in files on GPFS, it makes backup and recovery easier.

- The possibility to use the AUTOEXTEND feature of the Oracle tablespaces, similar to a JFS/JFS2 installation

- Scalability

  Since HACMP CLVM is not required, the maximum number of nodes in this configuration can be up to 32 nodes (8 with SSA).

Detailed information about this configuration is discussed in the following chapters.

3

Planning and implementation

In this chapter we describe how to configure an Oracle9*i* RAC cluster we used in our test environment. The main characteristics of our configuration were:

- A pSeries cluster comprising four nodes
- GPFS on top of an RPD cluster

We further discuss details about the hardware architecture and the software stack, as follows:

- Cluster nodes setup
- Network setup
- Storage subsystem setup

For each of the software layers, we present the required steps for package installation and configuration. This includes the operating system (AIX), the cluster infrastructure (RSCT), the cluster file system (GPFS), and the cluster enabling software (HACMP).

3.1 Configuration objectives

In the first release of Oracle9*i* RAC for IBM @server pSeries, only the concurrent RAW device configuration was supported. This configuration has a performance advantage over the file system implementation, but the administrative tasks that both system administrator and database administrator have to deal with are far more complicated.

In current version, 9.2.0.3, Oracle9*i* RAC supports the IBM cluster file system (GPFS) as shared datafile storage, which makes administration tasks almost similar to the standalone databases using “classic” file system storage.

For this book, we designed and tested an Oracle9*i* RAC cluster database on a GPFS cluster file system. We used the cluster file system for storing both Oracle9*i* RAC code (binaries and configuration files) and datafiles.

There are two main features to consider in an Oracle9*i* RAC cluster database implementation, features not available for the standalone database implementation:

- High availability
- Scalability

A comprehensive set of high availability tests is described in Chapter 6, “High availability test scenarios” on page 169. Our goal was to test the Oracle9*i* RAC high availability features in conjunction with IBM pSeries clusters.

In real life, application scalability may present several design and implementation issues that we also wanted to test, by configuring a cluster with more than two nodes. The reason: Most of the existing Oracle9*i* RAC implementations comprise only two nodes.

By implementing a four-pSeries node cluste,r we wanted to identify design and administrative issues this complex configuration may pose, especially when growing from two to four nodes.

3.2 Hardware architecture

The hardware platform used for our tests was composed of the following:

Nodes

We implemented a configuration consisting of four nodes:

- Two logical partitions (LPARs) in an IBM p690. Each LPAR has four processors and 8 GB of random access memory (RAM).
- Two IBM p660 6H1. Each node has four processors and 4 GB of RAM.

All the servers are a 64-bit CPU (mandatory for Oracle9*i* RAC) and PCI I/O architecture.

Networks

- One 2-Gbit Ethernet for Oracle9*i* RAC interconnect
- One 100-Mbit Ethernet for GPFS
- One 100-Mbit Ethernet for client and administrative network
- Six point-to-point serial non-IP networks (for HACMP use)

Storage

We chose to implement a Storage Area Network (SAN) architecture consisting of:

- 210 GB disk space formatted in RAID-5 in one ESS 2105-800. The ESS is connected to the SAN using two 2-Gbit Fibre Channel paths.
- Each node has two 2-Gbit 64-bit PCI FC adapter 6228s.
- One 2032-064 McData Enterprise Channel Director.

For an overview of the configuration used in our environment, see Figure 3-1.

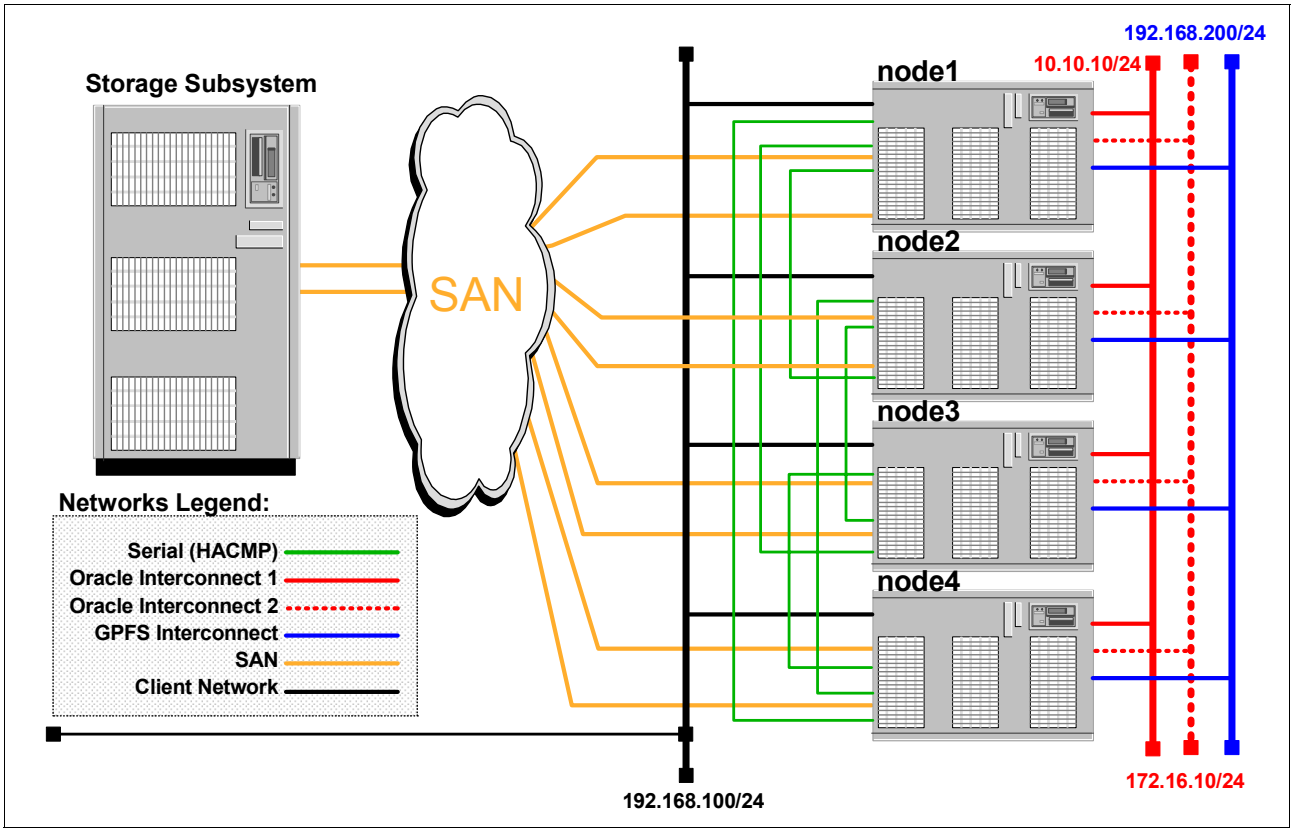


Figure 3-1 Hardware architecture

3.3 Software architecture

Using the hardware components presented in 3.2, “Hardware architecture” on page 22, we implemented an Oracle9*i* RAC based on GPFS as a shared storage solution.

For other possible configurations, refer to Chapter 2, “Typical Oracle9i RAC configurations” on page 7.

3.3.1 Oracle9*i* RAC on an RPD-based GPFS cluster

We decided to configure a 4-node GPFS cluster based on RSCT Peer Domain Cluster (RPD). All nodes in the cluster are GPFS servers and clients (a GPFS server has direct connection to the storage subsystem, a GPFS client mounts the file system). The characteristics of our implementation were:

- Oracle9*i* RAC datafiles and code stored on a shared file system, provided by GPFS.

- Multiple access paths to the ESS storage subsystem (for availability and performance).
- GPFS uses a 4-node RPD cluster.
- An HACMP/ES 4-node cluster is configured to provide Oracle9*i* RAC with cluster membership and event management information.

GPFS

GPFS needs a peer domain cluster layer to operate properly. This layer can be provided by either HACMP/ES or RSCT (RSCT peer domain cluster or RPD).

Note: In version 2.1, GPFS no longer requires HACMP.

In our test environment, we configured GPFS on top of an RPD cluster. This means that the HACMP cluster and GPFS/RPD clusters are independent. HACMP also provides a “GPFS integration feature” which adds SMIT menus for configuring a GPFS cluster in the same node set as the HACMP cluster.

We chose this setup because we wanted to be able to manage GPFS independent of the HACMP cluster. This allows an RPD cluster and an HACMP/ES cluster to coexist in the same node set. If you choose to configure the GPFS cluster using the HACMP “GPFS integration feature”, you will not be able to perform further manual changes to the GPFS configuration. Only HACMP SMIT menus are allowed for changing and managing GPFS clusters. For more information about the GPFS configuration, refer to 3.10.2, “GPFS cluster configuration” on page 52.

It is also possible to configure GPFS in an HACMP cluster. In this case, we do not have to explicitly define a peer domain cluster using RSCT commands. The cluster is defined and maintained automatically by HACMP, for both Oracle and GPFS.

Although possible, we do not provide detailed information about this configuration.

Figure 3-2 shows all the software layers for our cluster configuration (GPFS on top of RPD cluster).

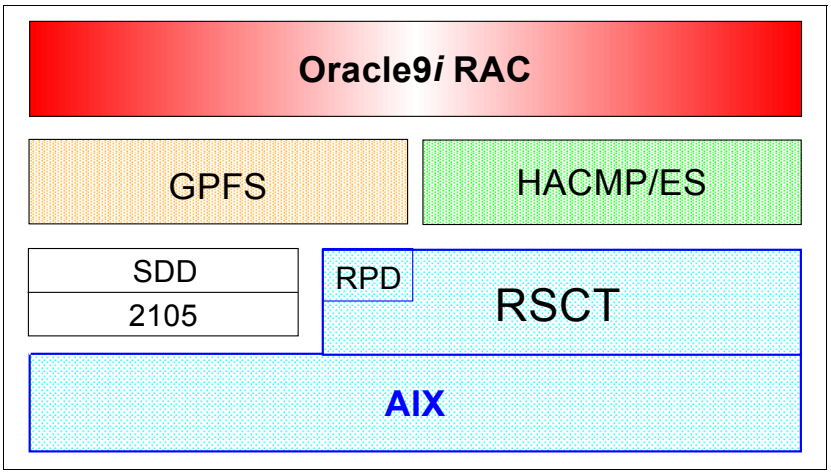


Figure 3-2 GPFS on an RPD cluster software architecture

AIX layer - The AIX operating system includes the RSCT cluster software, which provides for building the RPD clusters. For more information about RSCT, refer to “RSCT 2.3.1” on page 31.

2105 layer - For the database storage, we chose to put the datafiles and the Oracle9*i* RAC code on an Enterprise Storage Server (ESS), model 2105-800. To recognize this storage unit, the appropriate drivers must be installed on all the nodes. This step is detailed in 3.9.2, “Configuring logical disks” on page 46.

SDD layer - Subsystem Device Driver, also known as Data Path Optimizer (DPO). The nodes must be connected to the SAN through at least two Fibre Channel adapters. The goal is to provide high availability (two different paths to access the data) and load balancing (the I/O load is spread over all the available paths). When enabled, it provides the vpath entries, which are used instead of regular hdisks. For detailed installation and configuration, refer to 3.9.5, “ESS Subsystem Device Driver setup” on page 48.

GPFS layer - General Parallel File System is the IBM cluster file system. It provides a common file system space that allows database administrators to have the simplicity of the file-based storage (versus RAW devices) when running Oracle9*i* RAC.

With GPFS it is also possible to have only one Oracle9*i* code (binaries and configuration files) installation, common to all nodes in the cluster, thus simplifying software maintenance and backup operations.

HACMP/ES layer - (High Availability Cluster Multi Processing/Enhanced Scalability). This cluster layer is mandatory for Oracle9*i* RAC (if nodes are not part of a Cluster 1600 managed by PSSP). HACMP provides Oracle with cluster membership and event management information. It also provides the “private” networks used by Oracle9*i* RAC for interconnect traffic. Oracle9*i* RAC relies on network information provided by HACMP.

HACMP also provides event management functions, used by Oracle9*i* RAC to retrieve information about cluster status. No other high availability function of HACMP is involved in building an Oracle9*i* RAC cluster.

In our configuration, there is no IP address takeover (IPAT) defined. Oracle9*i* RAC provides its own high availability methods, and does not require IPAT. Moreover, IPAT must *not* be configured for any of the networks subject to use for interconnect traffic.

Note: Oracle9*i* RAC does not use the IPAT functions provided by HACMP. The IP addresses used by Oracle9*i* RAC on the server (interconnect and client networks) must remain unchanged. Moreover, it is not recommended to define any takeover resources on the same cluster nodes on which Oracle9*i* RAC runs.

For more information on how the Oracle interconnect network selection mechanism works, refer to 3.10.3, “HACMP 4.5 installation” on page 59.

Oracle9*i* RAC layer - (Real Application Clusters). This is the database application that runs on top of all the previously described layers.

3.4 Network architecture

The critical aspect of our design concerns the availability and performance of the Oracle9*i* RAC interconnect network. For our environment we designed a networking environment for two basic requirements:

- High availability
- High performance

3.4.1 Networking design

We decided to configure five logically and physically separated networks for:

- Client and administrative network
- First Oracle9*i* RAC Interconnect
- Second Oracle9*i* RAC Interconnect
- GPFS Interconnect (token manager traffic) network
- Serial (non-IP) network

Figure 3-3 shows the networks in our testing environment.

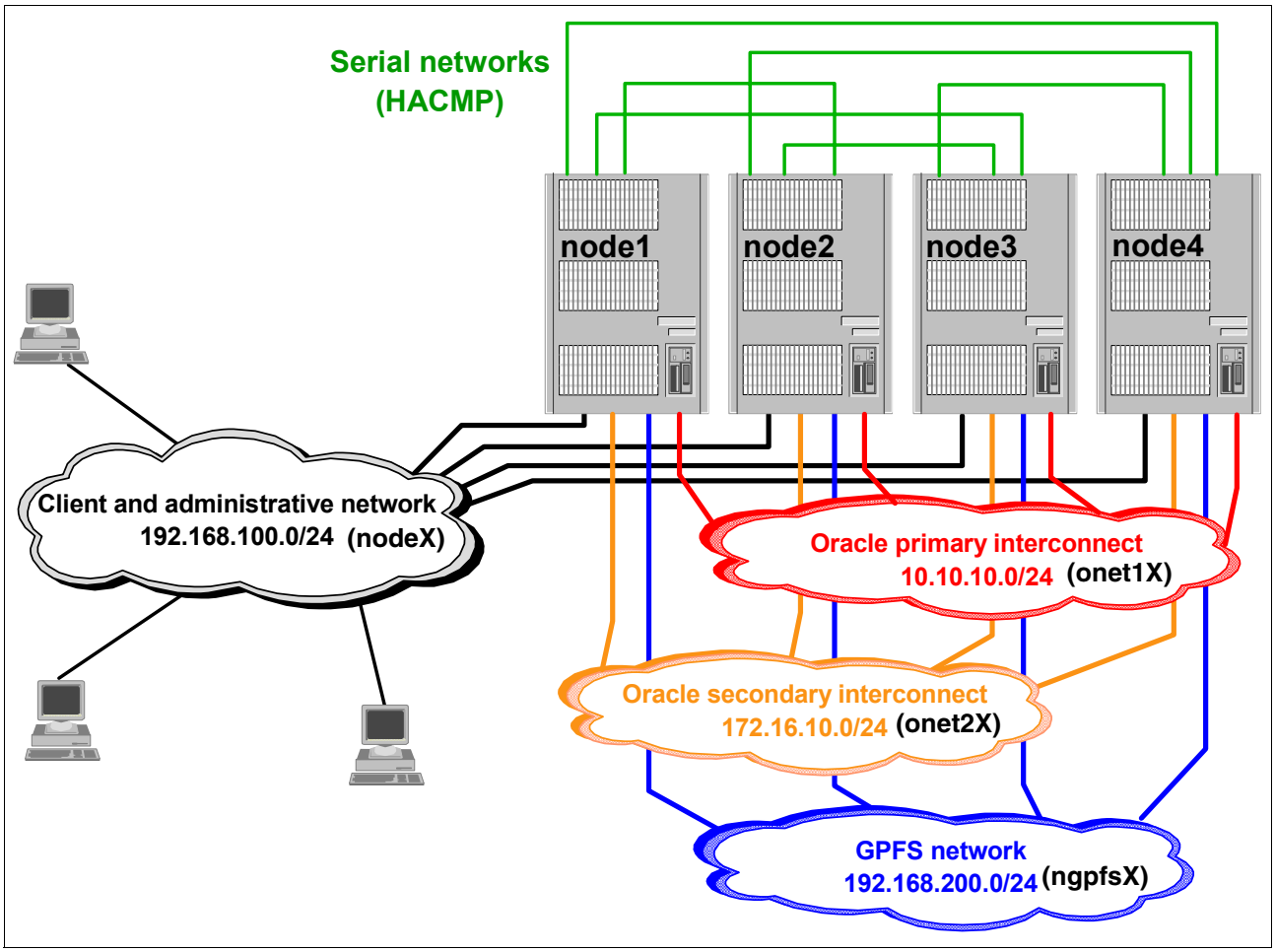


Figure 3-3 Network topology

3.4.2 Client and administrative network

This network is used for administration purposes and for the Oracle clients to connect to the RAC instances on the cluster.

For convenience, the IP labels assigned on this network are the same as the node hostnames (shown with “uname -n” - node1, node2, node3, node4).

Also, the HACMP node names as well as the Oracle node names are set to the same values: node1, node2, node3 and node4.

The load of this network may vary, depending on the number of simultaneously connected clients and the type of workload they are running. We recommend a normal 100-Mbit Ethernet, because, in a worst case scenario, when both Oracle private interconnects are down, this network may be used by Oracle for interconnect traffic (cache fusion data). When the private networks are restored and become available, interconnect traffic will be rerouted back over these networks.

3.4.3 Oracle interconnect networks

Oracle9*i* RAC uses the interconnect network for cache fusion purposes. Data blocks are exchanged between nodes, and service messages are routed through this way.

Interconnect network performance requirements

This is the key factor for the overall RAC database performance. These networks must meet two criteria: large bandwidth and low latency.

- Gbit Ethernet is the minimum recommended for reasonable performance.
- SP Switch or SP Switch2 are recommended in an SP environment.

The network latency is also important, because a query on one node could wait for a data block owned by another node, and to be sent through the interconnect.

See also 3.7.3, “Tuning network options” on page 38 for recommendations about network parameters.

High availability enabled

In normal operation, Oracle9*i* RAC uses a single interconnect link. On an AIX with HACMP platform it is possible, for high availability purposes, to configure backup links. In our environment we chose to configure two network links for this purpose.

If Oracle9*i* RAC loses network communication with other instances, the instances that lose communication are shut down. Thus we strongly recommend to double the Oracle interconnect networks. Along with HACMP/ES, Oracle9*i* RAC can handle up to three interconnect networks, two “private”, and one “public”. For more information on the interactions between HACMP and Oracle and these networks, see “How HACMP networks are used by Oracle” on page 78.

No load balancing

Multiple interconnect links may be established for redundancy and fault tolerance purposes. In an HACMP environment, even with two “private” networks, there is no load balancing between the interconnect networks. Only one will be used at a time.

If the primary network fails, Oracle9*i* RAC will detect it, and fail over to the second one. When the primary network comes back, all the cache fusion load will be routed to it again. While the primary interconnect network is operating normally, the secondary is used only for HACMP heart beat traffic.

In our environment, IP labels used for identifying the interconnect networks are:

- onet11, onet12, onet13, and onet14 for the primary interconnect network (10.10.10.0/24)
- onet21, onet22, onet23, and onet24 for the secondary interconnect network (172.16.10.0/24)

These two networks are dedicated for Oracle interconnect (cache fusion) traffic, and must be defined in HACMP and labeled as “private”.

For host naming details see Example 3-6, “The /etc/hosts file” on page 35.

3.4.4 GPFS network

For GPFS traffic we implemented a separate network. Since Oracle is responsible for maintaining locks between instances, there is no heavy data traffic over this network.

Read and write operations are performed via Storage Area Network (SAN). This network is for RSCT heart beat and GPFS metadata traffic only. A 100-Mbit Ethernet network is considered enough for this type of load.

Also, due to our decision to implement GPFS over an RSCT peer domain cluster (RPD), this network must *not* be defined is HACMP configuration.

The IP labels used for this network are ngpfs1, ngpfs2, ngpfs3 and ngpfs4.

3.4.5 Serial (non-IP) network

This network has to be implemented to allow HACMP to differentiate between a network failure and a TCP/IP subsystem failure. This avoids the “split brain” situation, allowing HACMP to make correct decisions in such cases.

Each node must have a direct serial link to all the others. While only one cable is enough for a 2-node cluster, the serial topology is more complicated with a higher number of nodes.

A non-IP network can be RS232, target mode SSA, and target mode SCSI. Multiport serial adapters are required to configure serial links. A non-IP network for HACMP is not mandatory for clusters without IPAT, but highly recommended.

Note: Since in this cluster configuration there are no resources or resource groups configured, special customization is required for HACMP to act in a “split brain” situation.

3.5 Storage subsystem architecture

All of the cluster nodes have internal SCSI disks, used for storing the operating system and additional software.

For the database files, we connect to an Enterprise Storage Server ESS, Model 2105-800, using a Storage Area Network. This storage is available via SAN to all cluster nodes.

Several aspects should be considered when planning the storage for such an environment:

- Storage must be protected to prevent data loss from a physical disk drive failure. This protection can be achieved using hardware RAID. The storage space configured in ESS-800 is configured as RAID5.
- Each node is connected to the ESS by two physical links. A special device driver, designed for ESS, provides load balancing as well as high availability. This is called Subsystem Device Driver (SDD).
- ESS has a fully redundant internal structure (built around a 2-node pSeries cluster, and using SSA storage). The two Fibre Channel links must be connected to different bays. Bays 1 and 2 are connected to ESS internal cluster1, bays 3 and 4 to ESS internal cluster2.

- When designing a SAN network, use two FC switches, or at least a fault tolerant switch with two separated zones. It is recommended to use switch separation or zoning to avoid multiple configuration of the same devices when using the SDD.

Figure 3-4 shows the topology of a SAN with no single point of failure. GPFS-supported storage types are IBM ESS and FAStT models, and SSA. For non-IBM storage, contact the manufacturer for GPFS compatibility and support information.

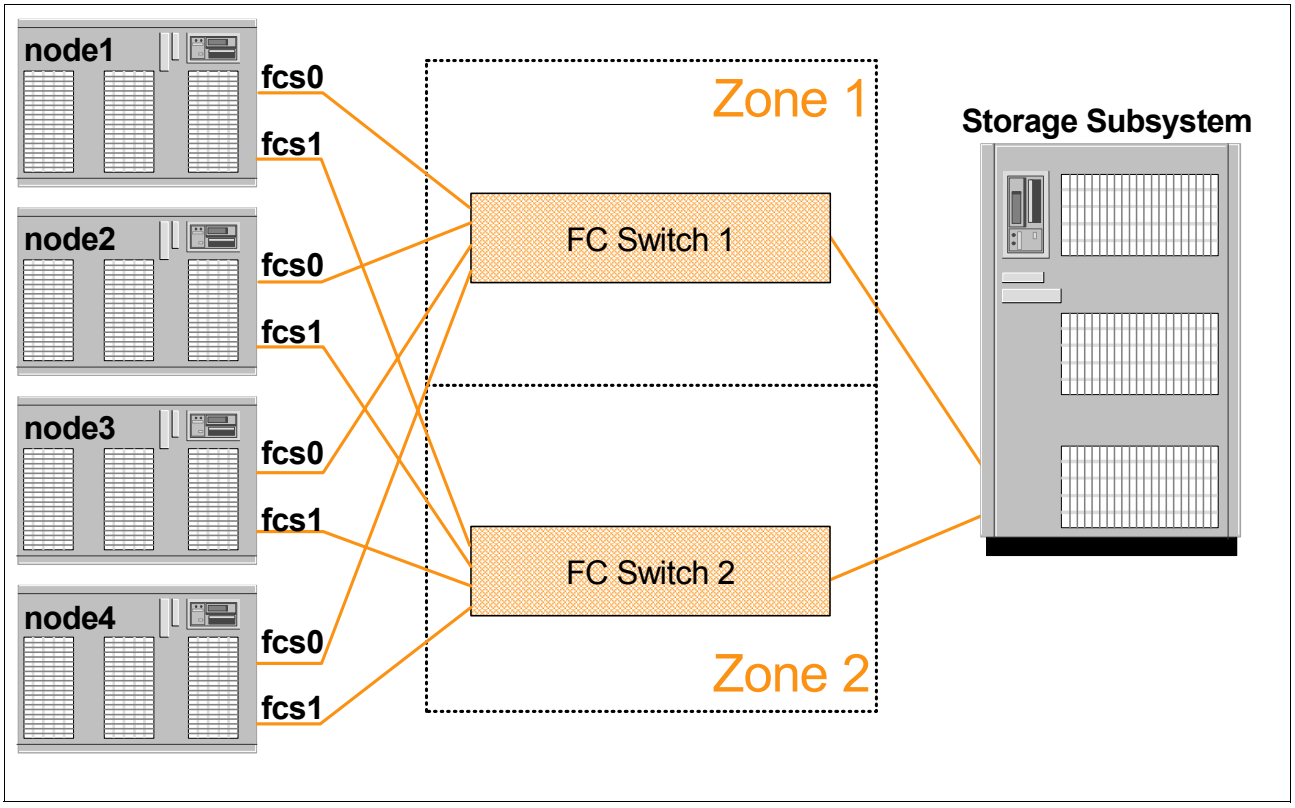


Figure 3-4 SAN Fibre Channel architecture

3.6 Node installation and configuration

Due to software complexity in our environment, careful planning with detailed steps were needed prior to implementation. The aspects you should consider when planning are related to basic OS configuration (AIX), several cluster layers (HACMP, RPD, GPFS, RAC), and a careful storage layout (SAN, ESS etc.). All the previously mentioned aspects are determined by the database type you plan to implement and use.

The OS installed on our IBM pSeries nodes is AIX 5L, version 5.2 ML01. The RSCT package, included in AIX, is installed together with the base OS. This makes it possible to configure an RSCT peer domain cluster (RPD) to be used by GPFS.

For a quick overview, a detailed check list is provided in 3.11, “Check list” on page 79.

3.6.1 AIX 5.2 ML1

The version of AIX used for our tests is AIX 5.2 Maintenance Level 01 (a.k.a. AIX 5.2B). We recommend that you use a Network Install Manager Master, since it might be faster to install and configure one node, and then creating a system backup image to be used for the other nodes in the cluster. This shortens the overall configuration time and also ensures a coherent software environment, since software maintenance is performed from a single point of control (the NIM Master).

In a cluster environment, it is a strongly recommended to have all the software packages at the same level on all the nodes inside the cluster. Using different versions may result in different behavior and unexpected operation (if not errors).

Note: For easing cluster-wide administrative tasks, consider using the dsh tool, which allows distributed command execution on all the nodes in the cluster, as described in “Setting up the dsh tool” on page 37.

File sets

To determine the current operating system version and maintenance level, issue the following command:

```
{node1:root}/-> oslevel -r
5200-01
```

A complete list of the base operating system filesets used in our environment is shown in Appendix A, “Operating system fileset levels” on page 197.

3.6.2 APARs/PTFs

If the installation documentation recommends a specific APAR (or PTF), you can verify whether the required APAR is installed on your system. For example:

```
{node1:root}/-> instfix -ik IY30887
    All filesets for IY30887 were found.
```

To download an APAR or PTF that is not found on your system, use:

```
https://techsupport.services.ibm.com/server/fixes
```

Note: Always check the release notes for the software you plan to install, and also check the Web for any last minute announcements.

AIX documentation

Documentation may be configured on your local system or you can access it online at:

```
http://www16.boulder.ibm.com/pseries/en_US/infocenter/base/aix.htm
```

Latest release notes for AIX can be found at:

```
https://techsupport.services.ibm.com/server/aix.fdc
```

Java

The list of Java fileset levels is shown in Appendix A, “Operating system fileset levels” on page 197.

RSCT 2.3.1

The Reliable Scalable Cluster Technology version that comes with AIX 5.2B is RSCT 2.3.1.

Refer also to the following RSCT publications:

IBM Reliable Scalable Cluster Technology for AIX 5L:Administration Guide, SA22-7889
IBM Reliable Scalable Cluster Technology for AIX 5L: Messages, GA22-7891
IBM Reliable Scalable Cluster Technology for AIX 5L:Technical Reference, SA22-7890
RSCT Group Services Programming Guide and Reference, SA22-7888
RSCT Event Management Programming Guide and Reference, SA22-7354
RSCT First Failure Data Capture Programming Guide and Reference, SA22-7454

You can view or download the PDF files from the following links:

```
http://www.ibm.com/servers/eserver/pseries/library/clusters/aix.html
http://www.ibm.com/servers/eserver/pseries/library
http://www.ibm.com/shop/publications/order
```

A complete list of the RSCT fileset is shown in Appendix A, “Operating system fileset levels” on page 197

3.6.3 AIX 5L 32/64-bit kernel considerations

Oracle9*i* RAC requires a 64-bit HW architecture, so we decided to use AIX 5.2 with a 64-bit kernel and JFS2 for our file systems. Note that Oracle9*i* RAC can run on either a 64-bit or a 32-bit kernel. If you do not plan to use any other Oracle Application Server on the same nodes (like Oracle Applications11*i,* which runs only on a 32-bit kernel), we recommend a full 64-bit OS configuration.

The following command returns the HW type (32bit or 64bit):

```
{node1:root}/-> getconf HARDWARE_BITMODE
64
```

Note that this command is valid for AIX 5.2 only.

To check the running kernel version, in AIX5L issue:

```
{node1:root}/-> getconf KERNEL_BITMODE
64
```

To switch between 32-bit and 64-bit kernels, in AIX 5L, as *root* user, issue:

```
# ln -sf /usr/lib/boot/unix_64 /unix
# ln -sf /usr/lib/boot/unix_64 /usr/lib/boot/unix
# bosboot -ad /dev/hdisk0 (or your ipl device, if different from hdisk0)
```

Note: To activate the new kernel, you *must* reboot the node.

Note also that the /usr/lib/boot/unix_mp file represents the 32-bit SMP kernel, the /usr/lib/boot/unix_up file the 32-bit uniprocessor kernel, and the /usr/lib/boot/unix_64 file the 64-bit SMP kernel. There is no 64-bit uniprocessor kernel.

3.6.4 File system considerations

Journaled File System (JFS) is the default for AIX 4.3.3 and earlier, as well as for AIX 5L installed with a 32-bit kernel.

Enhanced Journaled File System (also known as JFS2) is another native AIX journaling file system that has been introduced in AIX 5.1. JSF2 is the default file system for 64-bit kernel environments. Due to address space limitations of the 32-bit kernel, we recommend that you do not use the enhanced JFS2 in 32-bit environments.

Note: Although we do not recommend this, coexistence of JFS and JFS2 file systems in the same volume group is possible, because each file system type has its own logging device (jfslog or jfs2log).

3.6.5 Memory requirements

The minimum RAM requirement for the cluster nodes is 512 MB. Depending on your database implementation, this amount may not be enough. We recommend a minimum of 2 GB for all cluster nodes.

To check the physical memory, issue one of the following commands:

```
{node2:root}/-> lsattr -El sys0 -a realmem
realmem 8388608 Amount of usable physical memory in Kbytes False
{node2:root}/->  bootinfo -r
8388608
```

3.6.6 Paging space (swap) requirements

Paging space is required by the system as an extension to the Random Access Memory (RAM), and is managed by the kernel subsystem named Virtual Memory Manager (VMM). If the paging space is undersized, the system may either hang or have very poor response time.

On AIX, the paging space can be increased dynamically by adding RAW disk space to existing paging space logical volumes, or by creating new paging spaces (logical volumes). The amount of paging space you should configure depends on the amount of physical memory installed on the system, and the paging space requirements of your applications.

The system uses swap space only if it runs out of real memory. If the memory is of sufficient size, there is no paging and the page space can be small.

Workloads where the demand for pages does not fluctuate significantly perform well with a small paging space, whereas workloads likely to have peak periods of increased paging require enough paging space to handle the peak number of pages.

As a rule of thumb, for systems with more than 256 MB of RAM, use an initial setting for the paging space = (512 + (RAM - 256) * 1.25), with an upper limit of 32 GB. As a recommendation, try to spread the paging space over several physical disks. If possible, use

paging space on disks not heavily used and never use more than 20% of a physical disk for paging space.

Note that in a database environment, if the system memory is large enough (>2 GB), paging space may not be a concern, but if the system is running other applications, it may be necessary to increase the paging space. Excessive swapping activity in a database environment is an indication that the system real memory (RAM) may be undersized, thus increasing paging space may not solve this problem, considering database behavior (SGA is “pinned “ memory).

Use the **lsps -s** command to monitor global paging space use, and the **vmstat** command to monitor system memory, including paging activity. To increase the paging space, use the **smit pgsp** command.

The metric %Used in the output of **lsps -a** is typically less than 25% on a healthy system. Example 3-1 shows the paging space allocated on one node. As the %Used is only 1%, we see that the machine did not swap since the last boot.

Example 3-1 Paging space allocation

```
{node2:root}/-> lsps -a
Page Space      Physical Volume   Volume Group    Size %Used Active  Auto  Type
paging00        hdisk0            rootvg        4096MB     1     yes   yes    lv
hd6             hdisk0            rootvg        3584MB     1     yes   yes    lv
```

Undersizing the paging space is dangerous because, if the system is running out of memory, the system will not create new processes, and even start killing some of the existing ones, to free some memory. On the contrary, oversizing the paging will not affect a running system, other than wasting some disk space.

3.6.7 Temporary space

The Oracle Universal Installer (OUI) requires up to 800 MB of free space in a temporary directory.

To check the standard free temporary space available, use the following command:

```
{node1:oracle}/oracle/home-> df -k /tmp
Filesystem    1024-blocks      Free %Used    Iused %Iused Mounted on
/dev/hd3           917504    882136    4%      152     1% /tmp
```

You can specify another temporary space to be used during Oracle installation and software maintenance, thus avoiding oversizing the /tmp file system. In the Oracle profile, set the TEMP and TMPDIR environment variables (both used by Oracle) to the new location defined for this purpose.

```
export TEMP=/new_tmp
export TMPDIR=/new_tmp
```

For more information about the Oracle user environment, see 4.2, “Oracle9i RAC installation and configuration (on GPFS)” on page 88.

3.6.8 Environment and user settings

root user profile

For convenience, we recommend that you add the following directories to the root user search PATH variable.

Example 3-2 Setting the root user PATH variable

```
{node1:root}/-> cat /.profile
>>>>>>>>>>>>>Omitted Lines<<<<<<<<<<<<<<<<<<<<<<<
# For HACMP/ES
export PATH=$PATH:/usr/es/sbin/cluster:/usr/es/sbin/cluster/sbin
export PATH=$PATH:/usr/es/sbin/cluster/utilities:/usr/es/sbin/cluster/diag
export PATH=$PATH:/usr/es/sbin/cluster/etc
# For GPFS
export PATH=$PATH:/usr/lpp/mmfs/bin
# For dsh
export PATH=$PATH:/opt/csm/bin
>>>>>>>>>>>>>Omitted Lines<<<<<<<<<<<<<<<<<<<<<<<
```

For a complete dsh configuration, see also “Setting up the dsh tool” on page 37.

The Oracle user profile is described later in 4.2, “Oracle9i RAC installation and configuration (on GPFS)” on page 88.

System and user limits

Change the number of processes a user can run simultaneously (the system default is 128). Depending on the planned number of concurrent database clients, you have to change this to at least 512.

In our test environment we changed this number to 2000. This limit is usually set to a value too low for a database. We recommend to set this value to at least 2000, depending on the number of connections you plan to serve at the same time. This is valid system-wide, not depending on a user name.

For system parameter values, see Example 3-3.

Example 3-3 sys0 parameters in our environment

```
{node1:root}/-> lsattr -El sys0
SW_dist_intr false              Enable SW distribution of interrupts              True
>>>>>>>>>>>>>>>> Omitted lines <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
maxbuf       20                 Maximum number of pages in block I/O BUFFER CACHE True
maxmbuf      0                  Maximum Kbytes of real memory allowed for MBUFS   True
maxpout      33                 HIGH water mark for pending write I/Os per file   True
maxuproc     2048               Maximum number of PROCESSES allowed per user      True
minpout      24                 LOW water mark for pending write I/Os per file    True
>>>>>>>>>>>>>>>> Omitted lines <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
```

To change these values:

```
{node4:root}/-> chdev -l sys0 -a maxuproc=2048
sys0 changed
```

Also, AIX imposes some limitations for each user for the file size, the CPU time slice, the memory allocated, etc. Check Oracle documentation before setting these parameters.

Important: If possible, set these parameters only for Oracle users. Setting these parameters system-wide may affect other running software.

To display these limitations for each user, log in as that user and issue the following (we used *oracle* as user):

Example 3-4 Displaying user limits

```
{node4:oracle}/oracle/home-> ulimit -a
time(seconds)        unlimited
file(blocks)         unlimited
data(kbytes)         unlimited
stack(kbytes)        2097152
memory(kbytes)       unlimited
coredump(blocks)     2097151
nofiles(descriptors) 2000
```

You may change user limits through SMIT: **`smitty user`** fastpath.

These limits can also be modified by manually editing the /etc/security/limits file, as shown in Example 3-5. The default stanza applies to all users, unless a different value is set in a user's own stanza. A value of "-1" means unlimited.

Example 3-5 The /etc/security/limits file

```
{node2:root}/-> cat /etc/security/limits

default:
        fsize = 2097151
        core = 2097151
        cpu = -1
        data = 262144
        rss = 65536
        stack = 65536
        nofiles = 2000

root:

>>>>>>>>>>>>>>>>>>>>>> Omitted lines <<<<<<<<<<<<<<<<<<<<<<<<

oracle:
        fsize = -1
        data = -1
        stack = -1
```

3.7 Network configuration

3.7.1 Name resolution

Set up the /etc/hosts file and distribute it to all the nodes in the cluster. The configuration we used is shown in Example 3-6.

Example 3-6 The /etc/hosts file

```
{node3:root}/-> cat /etc/hosts
127.0.0.1              loopback localhost      # loopback (lo0) name/address
```

```
# Administrative network
192.168.100.71  node1
192.168.100.72  node2
192.168.100.73  node3
192.168.100.74  node4

# First oracle interconnect network
10.10.10.61     onet11
10.10.10.62     onet12
10.10.10.63     onet13
10.10.10.64     onet14

# Second oracle interconnect network
172.16.10.61    onet21
172.16.10.62    onet22
172.16.10.63    onet23
172.16.10.64    onet24

# GPFS network
192.168.200.61  ngpfs1
192.168.200.62  ngpfs2
192.168.200.63  ngpfs3
192.168.200.64  ngpfs4
```

To avoid name resolution time-out problems for our cluster, we set up the /etc/netsvc.conf file to contain this line:

```
hosts = local , bind
```

Note: Before further configuration, check the /etc/resolv.conf file and test the name resolution. *All* nodes in the cluster must resolve names identically.

3.7.2 Enabling remote command execution

Host equivalence

When working in a cluster environment, remote command execution should be enabled for coordinating actions and tasks between cluster nodes. For maintaining the software environment, HACMP, GPFS and Oracle require enabling of “r” commands. Although this may pose a security threat, this configuration has to be implemented, exposing systems security as little as possible.

HACMP needs remote execution enabled for cluster configuration and synchronization operations. The /.rhosts file must be set up for the root account on all HACMP nodes.

GPFS 2.1 also needs remote command execution for configuration and maintenance. Since we use a different network (not accessible to clients), the IP labels for GPFS must also be added to the /.rhosts file.

The IP labels used by Oracle9*i* RAC are usually the same as the HACMP node names. Oracle uses these HACMP IP labels to retrieve all the information about the cluster (node names, networks).

Oracle Universal Installer (OUI) installs the Oracle code on the local node, than uses remote copy commands (**rcp**) to distribute the code to the other nodes in the cluster.

Remote command execution is used by Oracle for centralized management operations (start/stop RAC instances etc.), thus we recommend that you set up the /etc/hosts.equiv file.

Note: If you are installing the Oracle software on a GPFS file system, the OUI installer automatically detects that the ORACLE_HOME directory is a cluster file system. The code is installed only once, and shared by all the nodes. In this case, host equivalency is not required for the Oracle user. On our platform, Oracle software is installed on a GPFS file system.

Setting up the /.rhosts file

This file is used for root users, and for the other users in case /etc/hosts.equiv is not found. Only the root user must have read/write permissions for this file, otherwise this is a security exposure. The permissions should be set to 600 (chmod 600 /.rhosts). See Example 3-7.

Example 3-7 The .rhosts file

```
{node3:root}/-> ls -l /.rhosts
-rw-------   1 root     system           68 Jun 23 09:47 /.rhosts
{node3:root}/-> cat /.rhosts
# this is mainly for dsh, HACMP and GPFS commnds
node1 root
node2 root
node3 root
node4 root

# GPFS network
ngpfs1 root
ngpfs2 root
ngpfs3 root
ngpfs4 root
```

Setting up the /etc/hosts.equiv file

When verifying permissions for remote commands, this file is used first for all non-root users. If either *group* or *other* have write permission set on this file, it will be ignored. The permissions should be set to 600 (rw-------). This file is used to enable Oracle users to execute remote commands. For our environment, see Example 3-8.

Example 3-8 The /etc/hosts.equiv file

```
{node3:root}/-> cat /etc/hosts.equiv
# @(#)61        1.5  src/tcpip/etc/hosts.equiv, tcpip, tcpip520 9/27/91 17:11:34
node1 oracle
node2 oracle
node3 oracle
node4 oracle
```

The /etc/hosts.equiv or ~/.rhosts files must be found on all the nodes that accept incoming remote commands.

Setting up the dsh tool

If your cluster contains more than two nodes, some of the administration tasks that have to be performed on all nodes become repetitive and time consuming. To automate these tasks and to avoid “missing” some operations on any of the nodes, we recommend that you use the distributed shell (dsh) tool.

To set up the dsh tool, follow these steps:

1. Check that the csm.dsh package is installed on your system. This is part of Cluster System Management (CSM), and comes with AIX 5.2.

   Note: A separate CSM license is required *only* if you plan to use a CSM cluster (CSM server package).

   ```
   {node2:root}/-> lslpp -lOu  csm.dsh
   Fileset                      Level  State      Description
   ----------------------------------------------------------------------------
   Path: /usr/lib/objrepos
     csm.dsh                    1.3.1.0  COMMITTED  Cluster Systems Management Dsh
   ```

2. On all the nodes, create a file containing the host names of the nodes you want to use for distributed command execution. Check that *group* and *other* have the right to read it. For example, we created the file /nodes, containing:

   ```
   {node2:root}/-> cat /nodes
   node1
   node2
   node3
   node4
   ```

3. On all the nodes, for all the users allowed to use dsh, modify the ~/.profile file by adding the following lines:

   ```
   export PATH=$PATH:/opt/csm/bin
   export WCOLL=/nodes
   ```

4. Also set the host equivalences. This is required for all the users to be able to use dsh. Refer to 3.8, “ESS Configuration” on page 40 to enable these equivalences.

5. Test the dsh:

   ```
   {node2:root}/-> dsh date
   node2: Mon Jun  2 11:50:29 EDT 2003
   node3: Mon Jun  2 11:50:29 EDT 2003
   node4: Mon Jun  2 11:50:29 EDT 2003
   node1: Mon Jun  2 11:50:29 EDT 2003
   ```

3.7.3 Tuning network options

Oracle9*i* RAC uses the User Datagram Protocol (UDP) for interprocess communications (over the interconnect network). You can tune UDP network settings on the nodes to improve Oracle performance. You can modify UDP buffering on AIX by changing the udp_sendspace and udp_recvspace parameters.

The udp_sendspace value must always be greater than Oracle9*i* RAC's db_block_size. Otherwise, one or more of the RAC instances will hang on startup.

Recommended values (used in our environment)

In AIX 5.2 there are two ways to set the network options:

AIX 5.2 “style”

The “new style” uses the /etc/tunables/nextboot file to set these parameters. This assumes that the pre520tune attribute of sys0 is set to “disable”:

```
{node2:root}/-> lsattr -El sys0 | grep pre52
pre520tune disable            Pre-520 tuning compatibility mode               True
```

In this case, you can set the parameters using the script shown in Example 3-9.

Example 3-9 Network options AIX 5.2 "style"

```
{node3:root}/-> cat /work/tune.sh
#!/bin/ksh
# Use the -p flag to specify that parameters should remain changed after
# a system reboot. Check in /etc/tunables/nextboot.
no -p -o udp_sendspace=65536
no -p -o udp_recvspace=655360
no -p -o tcp_sendspace=65536
no -p -o tcp_recvspace=65536
no -p -o ipqmaxlen=512
```

Pre-AIX 5.2 "style"

If you prefer the "traditional" way, you should change the sys0 attribute:

Example 3-10 Changing to pre250tune "style"

```
{node3:root}/-> chdev -l sys0 -a pre520tune=enable
sys0 changed
{node3:root}/-> lsattr -El sys0|grep pre520
pre520tune   enable        Pre-520 tuning compatibility mode                True
{node3:root}/->
```

In the traditional method, also named *pre520tune*, the **no** command flag **-p** is not available. When issuing the **no** command, changes are made for the current configuration, but are reset to the default values at the system restart. In order to make changes persistent after reboot, edit the file /etc/rc.net by adding the following lines:

```
if [ -f /usr/sbin/no ] ; then
/usr/sbin/no -o udp_sendspace=65536
/usr/sbin/no -o udp_recvspace=655360
/usr/sbin/no -o tcp_sendspace=65536
/usr/sbin/no -o tcp_recvspace=65536
/usr/sbin/no -o ipqmaxlen=512
fi
```

If problems occur, you should monitor for UDP socket buffer problems. You can do so by issuing the following command:

```
{node2:root}/-> netstat -p udp | grep "socket buffer overflows"
        0 socket buffer overflows
```

If the number of overflows is not zero, increase the value of the udp_recvspace parameter. The following command can be used to reset error counters before monitoring again:

```
{node2:root}/-> netstat -Zs -p udp
```

See also:

The "Monitoring and Tuning Communications I/O Use" section in the *"AIX5L Verison 5.2 Performance Management Guide"* for more information on AIX tuning parameters at:

```
http://publib16.boulder.ibm.com/doc_link/en_US/a_doc_lib/aixbman/prftungd/prftungd.pdf
```

TCP tuning on the clients for fast failover

One TCP/IP parameter on the client side influences the delay before the transparent application failover (TAF) occurs, in case of a client network failure on the server side.

Important: Check your TCP timers for all client platforms to be sure the Transparent Application Failover behaves as expected.

If you are using AIX clients to “drive” the database, check the tcp_keepidle parameter on the client. This parameter specifies the duration for which an idle TCP connection should remain active, measured in half-seconds. The default is 14400 (2 hours).

If you do not set this parameter to a reasonable value, let’s say 120 (1 minute), then in case of an instance or node failure, your client will wait for two hours before failing the TCP connection to another node.

To permanently change this parameter (if the pre520tune attribute of sys0 is set to “disable”):

```
{client:root}/-> no -p -o tcp_keepidle=<number of half-seconds>
```

If the pre520tune attribute of sys0 is set to “enable”, set this parameter also in the /etc/rc.net file.

Enable Gigabit Jumbo Frame

We also recommend that you enable the Jumbo Frame (Maximum Transmission Unit - MTU=9000) for the Gigabit Ethernet adapters. You can use the following SMIT panel procedures to enable the Jumbo Frame:

```
smitty eadap
Change / Show Characteristics of an Ethernet Adapter
(select the Gigabit Ethernet Adapter)
Transmit Jumbo Frames : yes
```

See also Chapter 6, “High availability test scenarios” on page 169 for a detailed description of the tests we performed in our test environment.

3.8 ESS Configuration

This task is performed using ESS Specialist, via any Web browser. Figure 3-5 on page 41 shows the ESS Specialist main screen. Use the Storage Allocation tab to proceed to the next step.

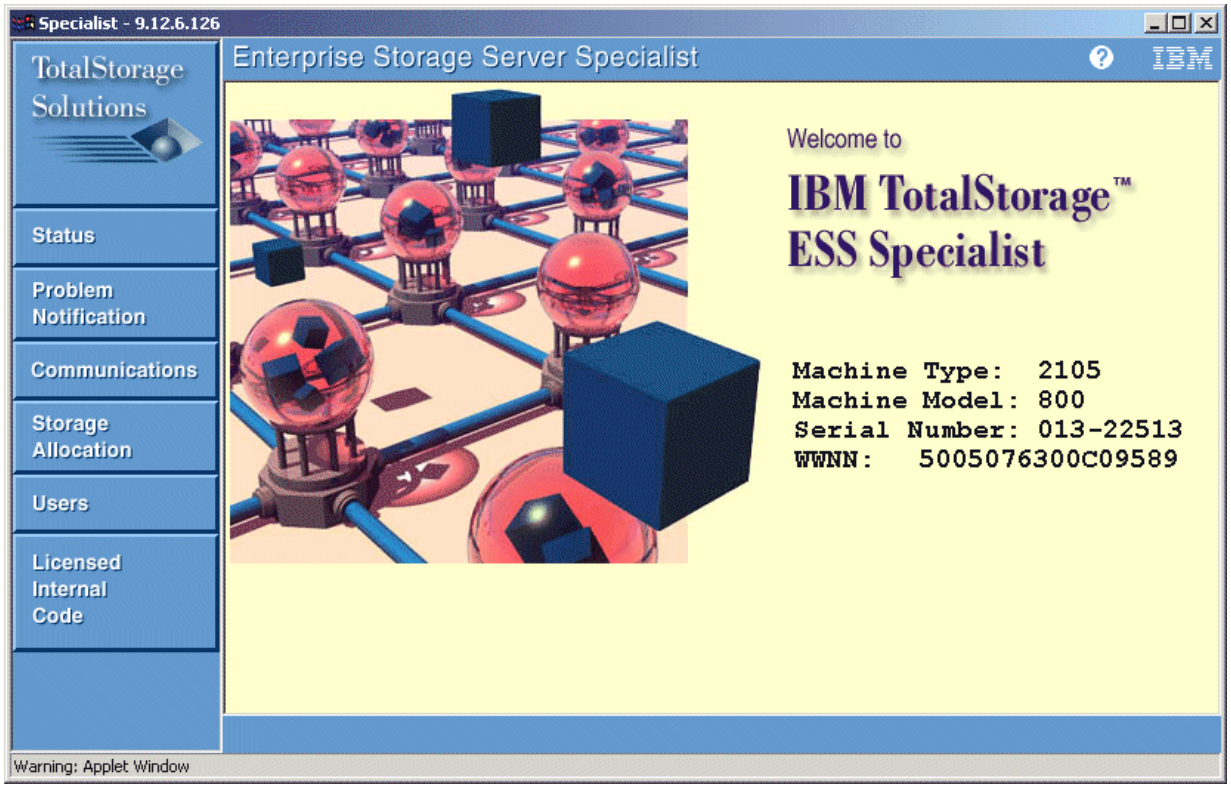


Figure 3-5 ESS storage specialist main screen

3.8.1 Configuring host adapter ports

In our platform, the storage subsystem is connected to the SAN through two optical fibers, connected to ESS bay2 adapter4, and bay3 adapter1. For high availability reasons it is recommended to use adapters belonging to different ESS clusters.

If two optical fibers are connected to adapters on the same ESS cluster, all the links will be lost in case this cluster fails, even though the ESS is still operating in degraded mode. See Figure 3-8 on page 44 for a graphical display of the ESS internal clusters and adapters.

Configure the FC adapters on the ESS side (see Figure 3-6 on page 42). Select the **Point to Point** (Pt2Pt) protocol for the adapters connected to the SAN switch.

Figure 3-6 ESS host adapter ports

3.8.2 Creating the hosts (on the storage side)

This step assumes that all the Fibre Channel adapters are installed on the AIX nodes, configured (fcs* devices in “available” status), and connected in the SAN with ESS.

In a storage area environment there are two ways to manage the access of different systems to the storage subsystem. Access is managed based on the World Wide Number (WWN) (each device connected to a SAN must have one), which is used to uniquely identify devices in a SAN environment. The two management methods are LUN Masking and Zoning.

- LUN Masking means restricting access at the storage level for specified hosts, identified by WWNs.
- Zoning means grouping some ports at the SAN switch level (similar to VLANs in a networking environment).

We are using LUN Masking at the ESS level to identify and limit access to the nodes to specific Logical Unit Numbers (LUN - SCSI terminology).

The first step is to define the host systems (our AIX cluster nodes, connected by their FC adapters) on the ESS. For ESS, each host is represented by a unique World Wide Number (WWN) belonging to each Fibre Channel adapter.

Make a list containing your node configuration with the corresponding FC adapters identified by their WWN (or IEEE Address). This list is useful when you configure the LUN masking on the ESS storage, and/or fabric zones (if you decide to use zoning).

To identify the WWN, check on the connector side of the adapter, or use the configuration information, as shown in Example 3-11 on page 45.

Each node is attached to the SAN by two Fibre Channel adapters. Thus, for each ESS link, we must define two hosts (both FC adapters) for the same AIX node.

For example, node1a represents the first node via the first link (first FC adapter - fcs0). Similarly, node1b is the same node, but via the second link (second FC adapter - fcs1).

Note: In ESS terminology, WWN is also referred to as World Wide Port Number (WWPN). WWN is also known as IEEE Address.

When creating a new host, you should select the appropriate WWN from a scroll list. If you don't see your WWN, refer to 3.12, "Troubleshooting" on page 80.

You should also select the two configured Fibre Channel ports as shown in Figure 3-7.

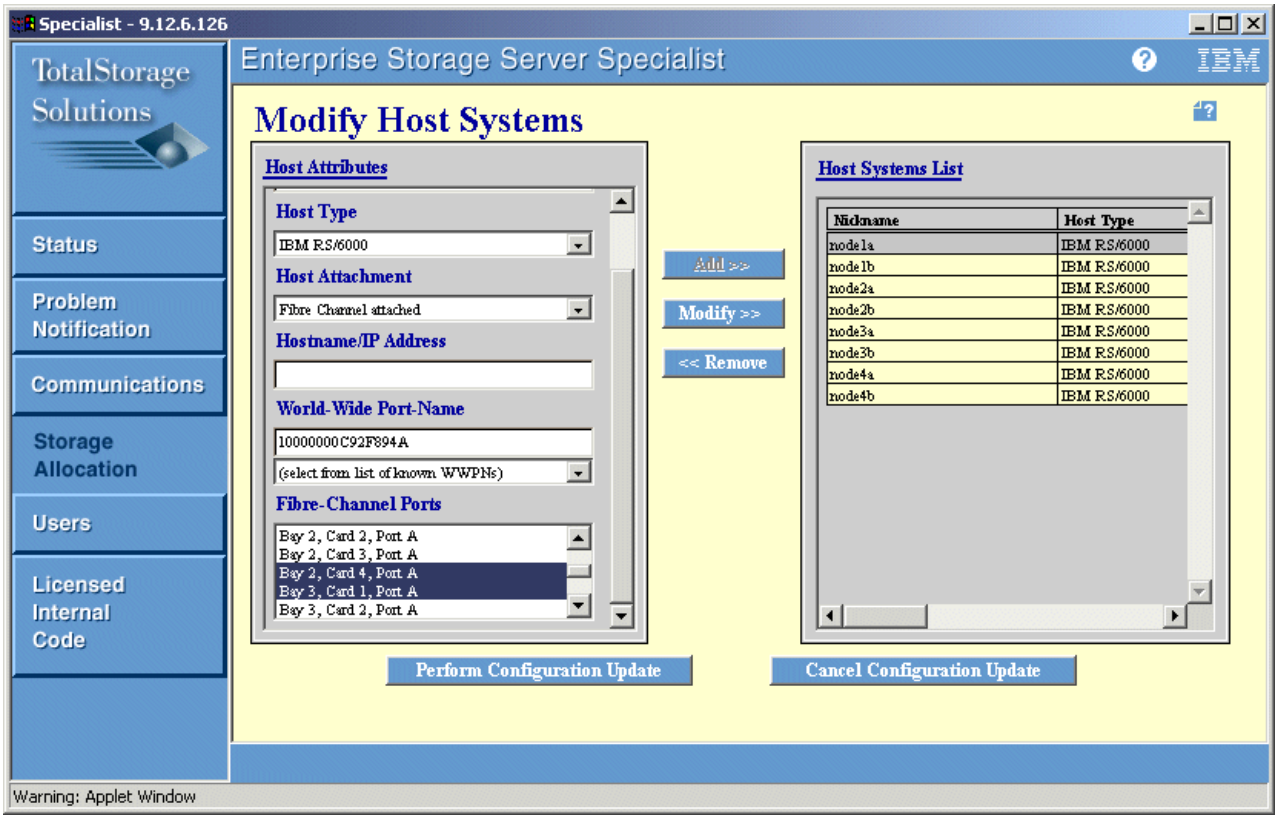


Figure 3-7 ESS host systems

Figure 3-8 shows the storage specialist main screen after node creation.

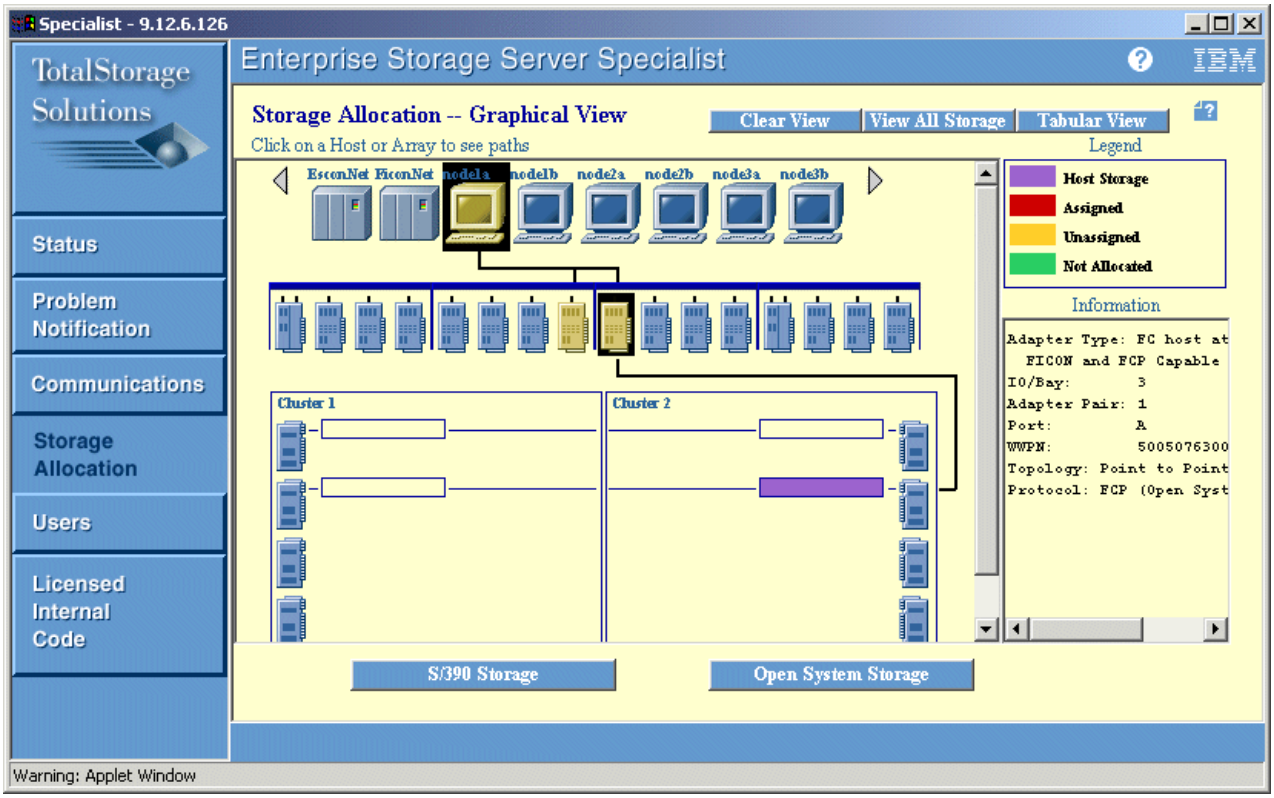


Figure 3-8 ESS main storage allocation

3.8.3 Creating the Logical Unit Numbers (LUNs)

We configured an ESS storage bay for use in our environment. This bay contains 8 x 36.4 GB disks configured in a (6+P+H) RAID5 array with fixed block, resulting in a total usable capacity of 216 GB.

We decided to split this array into several LUNs. When configured, each LUN is seen from the nodes as a logical disk (hdisk):

- 1x10 GB LUN for Oracle9 binaries and configuration files
- 4x46 GB LUNs for the Oracle9*i* RAC datafiles
- 1x10 GB LUN for administrative operations (exports, backups)
- 1x6.4 GB LUN spare

By the time it is created, each LUN is allocated only to a single node. Since we plan to use GPFS, we must configure all LUNs to be accessible from all nodes. Use the Modify Volume Assignments screen, as shown in Figure 3-9 on page 45.

After finishing the ESS configuration, close the ESS Specialist and proceed to configure logical storage configuration on the AIX nodes.

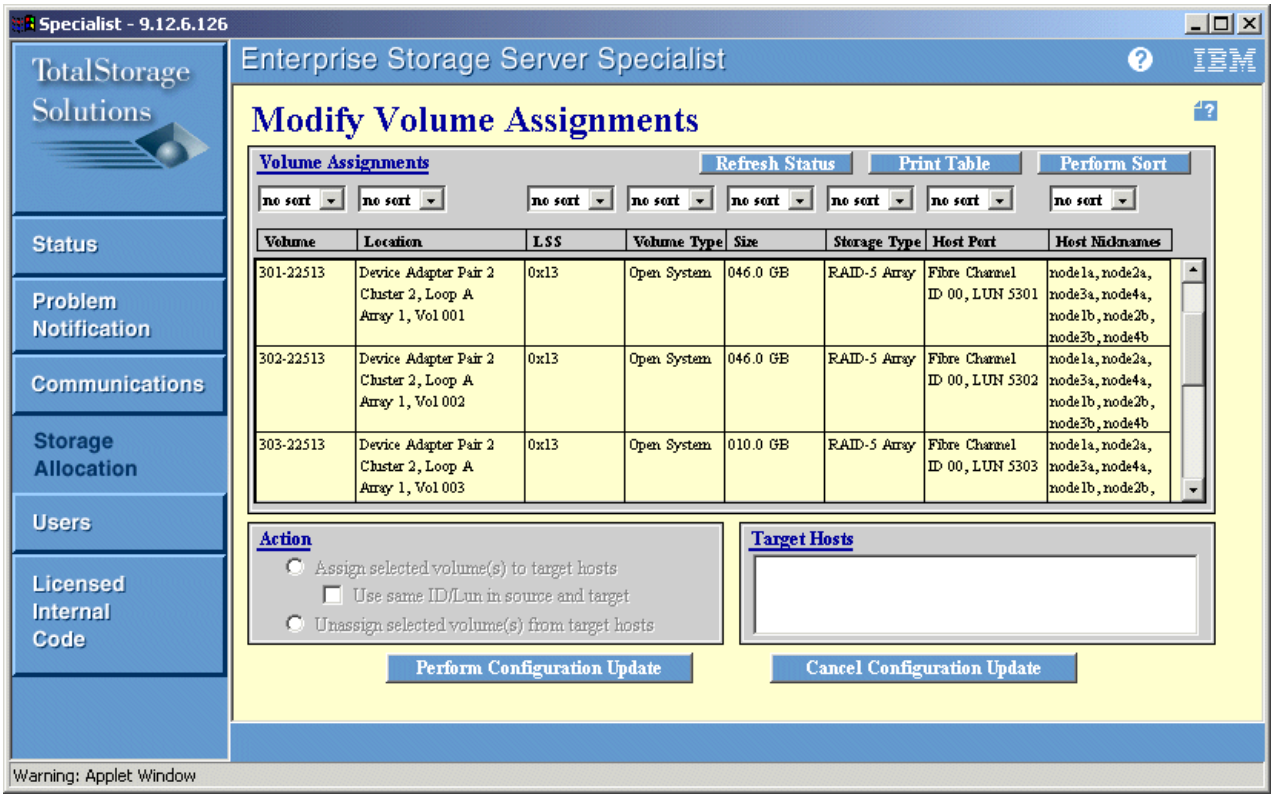


Figure 3-9 ESS LUN's tabular view

3.9 Cluster nodes SAN configuration

We have two Fibre Channel adapters per node, each on a separate PCI bus. This avoids possible PCI bottlenecks, and ensures redundant connection to the storage subsystem.

Check the FC adapters installed on the nodes (we used 2-GBbit Fibre Channel 64-bit Type 4W, FC #6228 adapters):

```
{node1:root}/-> lscfg | grep fcs
+ fcs1             U1.9-P1-I7/Q1                                  FC Adapter
+ fcs0             U1.9-P1-I1/Q1                                  FC Adapter
```

3.9.1 FC adapter microcode

Always check your FC adapter microcode level and resolve any issues with your IBM service representative.

Example 3-11 Fibre Channel adapter microcode (on the nodes)

```
{node1:root}/-> lscfg -vl fcs0
  fcs0             U1.9-P1-I1/Q1  FC Adapter

        Part Number.................00P2995
        EC Level....................A
        Serial Number...............1A238006CD
        Manufacturer................001A
        FRU Number.................     00P2996
        Network Address.............10000000C92F4E44 <--- World Wide Number
```

```
ROS Level and ID............02C03891
Device Specific.(Z0)........2002606D
Device Specific.(Z1)........00000000
Device Specific.(Z2)........00000000
Device Specific.(Z3)........02000909
Device Specific.(Z4)........FF401050
Device Specific.(Z5)........02C03891
Device Specific.(Z6)........06433891
Device Specific.(Z7)........07433891
Device Specific.(Z8)........20000000C92F4E44
Device Specific.(Z9)........CS3.82A1
Device Specific.(ZA)........C1D3.82A1
Device Specific.(ZB)........C2D3.82A1
Device Specific.(YL)........U1.9-P1-I1/Q1
```

To obtain the latest microcode level, check this link:

```
http://techsupport.services.ibm.com/server/mdownload/download.html#adapter
```

3.9.2 Configuring logical disks

There are two FC protocols, depending on whether an FC switch (fabric configuration) or an FC HUB (arbitrated loop) is used:

- The nodes are connected directly to the ESS, excluding any network equipment. Choose **Arbitrated Loop** (al protocol). This is the default value.
- The nodes are connected to a SAN, using a switch. Select **Point-to-Point** (pt2pt protocol).

Because our platform is using a switched SAN, the point-to-point protocol is selected on all nodes and ESS Fibre Channel adapters.

To view the actual value set on the FC adapter:

```
{node3:root}/-> lsattr -El fcs0 | grep init_link
init_link     al          INIT Link flags                                 True
```

To change this value, use “smitty devices”, select the FC Adapter menu, and change the field INIT Link flags. Or, you can use the command:

```
{node3:root}/-> chdev -l fcs0 -a init_link=pt2pt
```

If you are unsure about the way your nodes are connected to the disk subsystems, choose init_link=al. This setting first tries to detect a switch. If that fails, it changes to arbitrated loop. It is like an auto detect feature. The FC adapter will recognize the device it is connected to, that is, FC hub or switch.

Check whether all fiber cables are connected (nodes to FC switch, FC switch to ESS). This step assumes that the ESS storage configuration has been performed and checked.

For consistency, we recommend that you remove any previous definition of the FC adapters and their child devices, then run a fresh **cfgmgr**. On each node and for each FC adapter, issue:

```
{node3:root}/-> rmdev -Rdl fcs0
{node3:root}/-> cfgmgr
```

3.9.3 Enable Fast I/O Failure for FC adapters

AIX 5.2 supports the Fast I/O Failure feature for Fibre Channel devices in case of link events in a switched environment. If the FC adapter driver detects a link event (for example, a lost link between a storage device and a switch), the FC adapter driver waits a short period of time, about 15 seconds, to allow the SAN fabric to stabilize. At this point, if the FC adapter driver detects that the device is not on the SAN fabric, it begins failing all I/O requests at the adapter driver level. Any new I/O request, or future retries of the previously failed I/Os, are failed immediately by the adapter, until the adapter driver detects that the device has rejoined the SAN fabric.

Fast I/O Failure 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 that has existed in previous versions of AIX. Setting this attribute to `fast_fail` enables Fast I/O Failure, as shown in Example 3-12.

Example 3-12 FC adapter fast_fail option

```
{node1:root}/_> chdev -l fscsi0 -a fc_err_recov=fast_fail
```

The fscsi0 device is used for storage data transfer and is a child device of fcs0 (the FC adapter):

```
+ fcs0             U0.1-P1-I4/Q1              FC Adapter
* fscsi0           U0.1-P1-I4/Q1              FC SCSI I/O Controller Protocol Device
```

Fast fail logic is invoked when the adapter driver receives an indication from the switch that there has been a link event involving a remote storage device port via a Registered State Change Notification (RSCN) from the switch.

Fast I/O Failure may be desirable in situations where multipath software is being used. Setting fc_err_recov to `fast_fail` may decrease the I/O fail times due to link loss between the storage device and switch and allow faster failover to alternate paths.

In single-path configurations, especially in configurations with a single path to a paging device, the default `delayed_fail` setting is recommended.

Fast I/O Failure is only supported in a switched environment. It is not supported in arbitrated loop environments, including public loop.

We configured Fast I/O Failure since we were using the Subsystem Device Driver, two adapters per node and two paths to the storage subsystem.

The requirements for Fast I/O Failure support are the following:

- FC 6227 adapter firmware - level 3.22A1 or greater
- FC 6228 adapter firmware - level 3.82A1 or greater
- FC 6239 adapter firmware - all firmware levels

3.9.4 Dynamic tracking of Fibre Channel adapters

AIX 5.2 provides support for Dynamic Tracking of Fibre Channel devices. In previous AIX releases, the user was required to unconfigure the FC storage device and adapter device instances before making changes on the SAN that might result in an N_Port ID (SCSI ID) change of any remote storage ports.

If Dynamic Tracking for FC devices is enabled, the FC adapter driver will detect when the Fibre Channel N_Port ID of a device changes, and will reroute traffic designated for that

device to the new address, while the devices are still online. Examples of events that can cause an N_Port ID to change are moving a cable between a switch and storage device from one switch port to another, connecting two separate switches via an Inter-Switch Link (ISL), and possibly rebooting a switch.

Dynamic tracking of FC devices is controlled by a new fscsi device attribute, named `dyntrk`. The default setting for this attribute is `no`. Setting this attribute to `yes` enables dynamic tracking:

```
{node1:root}/-> chdev -l fscsi0 -a dyntrk=yes
```

Dynamic tracking logic is invoked when the adapter driver receives an indication from the switch that there has been a link event involving a remote storage device port.

3.9.5 ESS Subsystem Device Driver setup

In a high availability environment, there is a special device driver, designed for ESS, named Subsystem Device Driver (SDD). This device driver allows for redundant links and load sharing for storage traffic when multiple fiber connections exist between nodes and the ESS storage subsystem.

SDD comes as an AIX installable fileset, named ibm2105.rte. This has to be installed on all cluster nodes, even if not all nodes in the cluster have more than one FC adapter.

In our configuration, since each node is connected to the ESS using two optical cables, each disk can be accessed via any of the two paths. When SDD is installed, a virtual path is created. This virtual path represents the same storage space, but is accessible via both fiber links.

There are two versions of the SDD driver for AIX:

- ibmSdd_510.rte - This is suitable for non-HACMP configurations or for concurrent HACMP (HACMP/ESCRM).
- ibmSdd_510nchamp.rte - This has to be used in nonconcurrent HACMP environments.

In our environment, we are using HACMP, but not with concurrent disk access (locking is provided by GPFS and Oracle), thus we installed ibmSdd_510nchamp.rte.

See Appendix A, “Operating system fileset levels” on page 197 for more details on the fileset names and releases.

The latest SAN software can be downloaded from:

```
http://ssddom02.storage.ibm.com/techsup/webnav.nsf/support/2105
```

3.9.6 Configuring the virtual path devices

We recommend that you start with a “fresh” disk configuration, so it is a good idea to delete all previously configured FC adapters and their child (disk) devices.

On node1, we checked which disks are still defined:

```
{node1:root}/-> lspv
hdisk0          0022be2ab1cd11ac                    rootvg          active
hdisk1          0022be2a3d02ead0                    None
hdisk2          0022be2a4cbbafd8                    None
hdisk3          none                                None
```

These are the internal SCSI disk drives

```
{node1:root}/-> lscfg | grep disk
+ hdisk3           U1.9-P2/Z2-A8    16 Bit LVD SCSI Disk Drive (36400 MB)
+ hdisk2           U1.9-P2/Z1-A8    16 Bit LVD SCSI Disk Drive (36400 MB)
+ hdisk1           U1.9-P1/Z2-A8    16 Bit LVD SCSI Disk Drive (36400 MB)
+ hdisk0           U1.9-P1/Z1-A8    16 Bit LVD SCSI Disk Drive (36400 MB)
```

In order to include the ESS disks, run the configuration manager on each node:

```
{node1:root}/-> cfgmgr -v
```

Since ESS was configured with two host paths for each node (node1a and node1b), this results in two hdisks on the nodes. Actually, those two logical hdisks represent the same physical disk, accessed via the two ESS configured paths.

When the SDD driver package is installed, a third disk entry, named vpath* (virtual path) is automatically created. To benefit from the SDD capabilities, when configuring volume groups, use this virtual path disk entry instead of the regular hdisk. This provides for load balancing and high availability.

In normal mode, storage traffic is balanced over the two FC adapters installed. In case one of the two FC adapters fails for any reason, data access continues over the surviving adapter.

Check the new disk configuration discovered by cfgmgr:

Example 3-13 Listing the virtual path devices

```
{node1:root}/-> lsdev -Cs dpo
vpath0  Available              Data Path Optimizer Pseudo Device Driver <-+-
vpath1  Available              Data Path Optimizer Pseudo Device Driver   |
vpath2  Available              Data Path Optimizer Pseudo Device Driver   | Each vpath
vpath3  Available              Data Path Optimizer Pseudo Device Driver   | represents one
vpath4  Available              Data Path Optimizer Pseudo Device Driver   | physical LUN.
vpath5  Available              Data Path Optimizer Pseudo Device Driver   |
vpath6  Available              Data Path Optimizer Pseudo Device Driver <-+-
```

Example 3-14 Listing the FC attached disks (LUNs)

```
{node1:root}/-> lsdev -Cs fcp
hdisk4  Available 2V-08-01     IBM FC 2105800 <--+-
hdisk5  Available 2V-08-01     IBM FC 2105800    |
hdisk6  Available 2V-08-01     IBM FC 2105800    |
hdisk7  Available 2V-08-01     IBM FC 2105800    |
hdisk8  Available 2V-08-01     IBM FC 2105800    |
hdisk9  Available 2V-08-01     IBM FC 2105800    | These are the FC disks; there are only
hdisk10 Available 2V-08-01     IBM FC 2105800    | seven physical LUNs, but they are
hdisk11 Available 2v-08-01     IBM FC 2105800    | doubled because each node is connected
hdisk12 Available 2v-08-01     IBM FC 2105800    | to the storage using two FC adapters.
hdisk13 Available 2v-08-01     IBM FC 2105800    |
hdisk14 Available 2v-08-01     IBM FC 2105800    |
hdisk15 Available 2v-08-01     IBM FC 2105800    |
hdisk16 Available 2v-08-01     IBM FC 2105800    |
hdisk17 Available 2v-08-01     IBM FC 2105800 <--+-
```

Important: depending on the number of hdisks defined on a node prior to the ESS disk setup, the numbering of the new hdisks and vpaths may vary from one node to another. The only information you should rely on is the PVID (Physical Volume IDentifier) of the disk. This identifier is written on the disk, and retrieved by each node. It is the second column of the **lspv** command output.

To check the physical volumes, execute the following command:

Example 3-15 List of vpath devices created by SDD (extract)

```
{node1:root}/-> lspv
hdisk0          0022be2ab1cd11ac                    rootvg          active
hdisk1          0022be2a3d02ead0                    None
...
hdisk9          0022be2a31fa6b48                    None
...
hdisk16         0022be2a31fa6b48                    None
hdisk17         none                                None
vpath0          none                                None
vpath1          none                                None
vpath2          none                                None
vpath3          none                                None
vpath4          none                                None
vpath5          none                                None
vpath6          none                                None
```

To make these disks available to all the cluster nodes, type:

```
{node1:root}/-> chdev -l vpath5 -a pv=yes
{node1:root}/-> lspv | grep vpath5
vpath5          0022be2a31fa6b48                    None
Disk Name       PVID                                Volume Group
```

Note: To benefit from high availability and load balancing of SDD, only the vpath must be used for further LVM configurations (only vpaths should be used for volume groups).

.The SDD driver provides a set of commands to manage the virtual path devices:

Example 3-16 Datapath command arguments

```
datapath query adapter [n]
datapath query device [n]
datapath set adapter <n> online/offline
datapath set device <n> path <m> online/offline
datapath set device <n>/(<n1> <n2>) policy rr/fo/lb/df
datapath query adaptstats [n]
datapath query devstats [n]
datapath open device <n> path <n>
```

To see the correspondence between the hdisks and vpaths, use the following command:

Example 3-17 Datapath query device command

```
{node1:root}/-> datapath query device

Total Devices : 7

DEV#:   0  DEVICE NAME: vpath0  TYPE: 2105800   SERIAL: 30022513
POLICY:    Optimized
============================================================================
Path#      Adapter/Hard Disk          State     Mode     Select     Errors
    0          fscsi0/hdisk4           OPEN   NORMAL     253908          0
    1         fscsi1/hdisk11           OPEN   NORMAL     255351          0

DEV#:   1  DEVICE NAME: vpath1  TYPE: 2105800   SERIAL: 30122513
POLICY:    Optimized
```

```
============================================================================
Path#      Adapter/Hard Disk          State     Mode     Select     Errors
    0          fscsi0/hdisk5           OPEN   NORMAL       1749          0
    1         fscsi1/hdisk12           OPEN   NORMAL       1661          0

DEV#:   2  DEVICE NAME: vpath2  TYPE: 2105800   SERIAL: 30222513
POLICY:    Optimized
============================================================================
Path#      Adapter/Hard Disk          State     Mode     Select     Errors
    0          fscsi0/hdisk6           OPEN   NORMAL       1755          0
    1         fscsi1/hdisk13           OPEN   NORMAL       1773          0

DEV#:   3  DEVICE NAME: vpath3  TYPE: 2105800   SERIAL: 30322513
POLICY:    Optimized
============================================================================
Path#      Adapter/Hard Disk          State     Mode     Select     Errors
    0          fscsi0/hdisk7           OPEN   NORMAL        263          0
    1         fscsi1/hdisk14           OPEN   NORMAL        278          0

DEV#:   4  DEVICE NAME: vpath4  TYPE: 2105800   SERIAL: 30422513
POLICY:    Optimized
============================================================================
Path#      Adapter/Hard Disk          State     Mode     Select     Errors
    0          fscsi0/hdisk8           OPEN   NORMAL       1696          0
    1         fscsi1/hdisk15           OPEN   NORMAL       1670          0

DEV#:   5  DEVICE NAME: vpath5  TYPE: 2105800   SERIAL: 30522513
POLICY:    Optimized
============================================================================
Path#      Adapter/Hard Disk          State     Mode     Select     Errors
    0          fscsi0/hdisk9           CLOSE  NORMAL       1485          0
    1         fscsi1/hdisk16           CLOSE  NORMAL       1455          0

DEV#:   6  DEVICE NAME: vpath6  TYPE: 2105800   SERIAL: 30622513
POLICY:    Optimized
============================================================================
Path#      Adapter/Hard Disk          State     Mode     Select     Errors
    0         fscsi0/hdisk10           CLOSE  NORMAL          0          0
    1         fscsi1/hdisk17           CLOSE  NORMAL          0          0
```

3.10 Configuring a clustering infrastructure

As a remainder, the clustering infrastructure consists of a GPFS cluster defined on top of an RSCT peer domain cluster (RPD) and a separate HACMP cluster for Oracle9*i* RAC definition and management.

We defined a separate set of adapters (and IP subnetworks) for each cluster function (GPFS/RPD, HACMP/Oracle9*i* RAC). See 3.4, “Network architecture” on page 25.

3.10.1 RSCT Peer Domain (RPD) cluster

The RSCT Peer Domain (RPD) cluster is used as the infrastructure for our GPFS cluster and provides GPFS with node membership (topology and group services).

The RSCT commands used for configuring and maintaining an RPD cluster are provided with AIX.

Creating an RPD cluster

Before configuring the cluster, all the future cluster nodes have to be prepared for operating in a secure cluster environment (establishing trust between the nodes). For example, node preparation includes the PKI (Public Key Infrastructure) key pair generation for each of the nodes. These keys will be used for securing access to resources between cluster nodes. Then we have to choose a node that will be the “originator” for our cluster.

The following list shows the steps for configuring an RPD cluster:

1. The node from which you will issue the **mkrpdomain** command is called the originator node. We select node1 as our originator node. Run the **preprpnode** command on each cluster node:

   ```
   {node1:root}/-> preprpnode ngpfs1
   ```

2. Create a peer domain definition.

   After we establish trust between all the nodes, we can issue peer domain administration commands in originator node. Issue the following command on node1 to create the RPD cluster:

   ```
   {node1:root}/-> mkrpdomain gpfsrpd ngpfs1 ngpfs2 ngpfs3 ngpfs4
   ```

You also have the option to specify a node definition file by using the -f flag to point to a file that lists the node IP labels to be used for this RPD cluster. For example, we created the rpdnode.list file that contains all the nodes (IP labels of the adapters used for the RSCT heartbeat) for our RPD cluster, as shown in Example 3-18.

Example 3-18 RPD node list file

```
{node1:root}/-> # cat /gpfs_configure/rpdnode.list
# GPFS network IP labels - see /etc/hosts
ngpfs1
ngpfs2
ngpfs3
ngpfs4
{node1:root}/-> mkrpdomain -f /gpfs_configure/rpdnode.list gpfsrpd
```

3. Activate the peer domain:

   ```
   {node1:root}/-> startrpdomain gpfsrpd
   ```

4. You can use **lsrpdomain** to check the status of the peer domain, as shown in Example 3-19.

Example 3-19 RPD cluster status

```
{node1:root}/-> lsrpdomain
Name    OpState RSCTActiveVersion MixedVersions TSPort GSPort
gpfsrpd Online  2.3.1.0           No            12347  12348
```

After RPD cluster configuration and activation, you can proceed to GPFS configuration.

3.10.2 GPFS cluster configuration

GPFS fileset installation

Install the GPFS LPPs using **smitty installp** or the installp command line. Check for the latest package releases available for GPFS version 2.1. Some of the packages are mandatory, others are optional; for example, mmfs.gpfsdocs.data, which contains the GPFS

man pages, is not mandatory, although we suggest that you install it for getting the on-line help.

Note: When you install GPFS 2.1, it is mandatory to accept the electronic license agreement. When using **smit**, specify “yes” for the “ACCEPT new license agreements” field. If you fail to do so, the GPFS filesets will not be installed.

PTFs

The minimum requirement is U486402 - mmfs.base.cmds 3.5.0.2. In our environment we installed mmfs.base.cmds 3.5.0.6.

The complete list of fileset levels is shown in Appendix A, “Operating system fileset levels” on page 197.

Cluster layer for GPFS

In a previous GPFS version (1.5), the cluster layer could be provided by either HACMP or PSSP. Starting with GPFS 2.1, the RSCT Peer Domain (RPD) cluster can be used for the CPFS configuration. The RSCT cluster infrastructure is included in AIX 5.2, and does not require any additional licensing.

HACMP provides a GPFS integration feature, which allows users to configure the GPFS cluster in the same node set as the HACMP cluster using HACMP SMIT menus. At the time of writing, the GPFS inegration feature had been tested and validated only with GPFS 1.5. In our environment,since we used GPFS 2.1, we chose to configure the GPFS file system on top of an RPD cluster. Note that HACMP/ES provides a similar type of cluster (peer domain), which can be used for the GPFS cluster definition.

We decided to use an HACMP/ES cluster only for Oracle9*i* RAC, but you can choose to use the same cluster for GPFS as well, in case you do not need to define a separate RPD cluster.

Steps for creating a GPFS cluster

1. The set of nodes over which GPFS is defined is known as a GPFS cluster. Before creating the GPFS cluster, check if the previously defined gpfsrpd RPD cluster is up and running. For checking and troubleshooting, refer to 3.10.1, “RSCT Peer Domain (RPD) cluster” on page 51.
2. Prepare a node configuration file, NodeFile, containing the list of nodes and the cluster domain definition (either RPD, as in our configuration, or HACMP). The nodes in the GPFS cluster may be part of the RPD cluster or the HACMP cluster. The node list contains either the IP labels or the IP address that refer to the communication adapters which will be used for GPFS. IP aliases are not allowed.

   We created the following NodeFile file, named node.list, as shown in Example 3-20.

Example 3-20 GPFS cluster NodeFile

```
{node1:root}/-> cat /gpfs_configure/node.list
# GPFS network
ngpfs1
ngpfs2
ngpfs3
ngpfs4
```

   To prevent the loss of configuration data when the primary GPFS cluster data server goes down, we have to define the secondary GPFS cluster data server.

3. To create the GPFS cluster (on top of RPD), issue the following command:

```
{node1:root}/-> mmcrcluster -t rpd -p ngpfs1 -s ngpfs2 -n /gpfs_configure/node.list
```

To create the GPFS cluster on top of HACMP if not using RPD:

```
{node1:root}/-> mmcrcluster -t hacmp -p ngpfs1 -s ngpfs2 -n /gpfs _configure/node.list
```

Check the GPFS cluster status

4. After setting up the GPFS cluster, use **mmlscluster** to check the status, as shown in Example 3-21.

Example 3-21 GPFS cluster status

```
{node1:root}/-> mmlscluster
GPFS cluster information
========================
  Cluster id:  gpfs030521212516
  Remote shell command:      /usr/bin/rsh
  Remote file copy command:  /usr/bin/rcp

GPFS cluster data repository servers:
-------------------------------------
  Primary server:    ngpfs1
  Secondary server:  ngpfs2

Cluster nodes that are not assigned to a nodeset:
-------------------------------------------------
   1  ngpfs1      192.168.200.61   ngpfs1
   2  ngpfs2      192.168.200.62   ngpfs2
   3  ngpfs3      192.168.200.63   ngpfs3
   4  ngpfs4      192.168.200.64   ngpfs4
```

Create a GPFS nodeset

5. A nodeset is a group of nodes that all run the same level of GPFS and operate on the same file system. Within a GPFS cluster, the nodes can be divided into one or more GPFS nodesets. However, a node may only belong to one nodeset.

   Use the **mmconfig** command to create the GPFS cluster:

```
{node1:root}/-> mmconfig -n /gpfs_configure/node.list -A -C nodeset1 -D /tmp/mmfs -p 128M
```

   Then, use **mmlsnode** and **mmlsconfig** to list the configuration. Example 3-22 shows the **mmlsnode** command output.

Example 3-22 GPFS mmlsnode command

```
{node1:root}/-> mmlsnode -a
GPFS nodeset    Node list
-------------   -------------------------------------------------------
   nodeset1     ngpfs1 ngpfs2 ngpfs3 ngpfs4
```

   The pagepool parameter should be considered when creating the GPFS nodeset. Pagepool is the amount of pinned memory reserved for caching data read from disk. It is used for read-ahead and write-behind operations to increase performance. We set the initial value to 128 MB.

   Example 3-23 shows the **mmlsconfig** command output.

Example 3-23 GPFS mmlsconfig command

```
{node1:root}/-> mmlsconfig
Configuration data for nodeset nodeset1:
----------------------------------------
```

```
clusterType rpd
comm_protocol TCP
multinode yes
autoload yes
useSingleNodeQuorum no
pagepool 128M
dataStructureDump /tmp/mmfs
wait4RVSD no
group Gpfs.nodeset1
recgroup GpfsRec.nodeset1
useDiskLease yes

File systems in nodeset nodeset1:
---------------------------------
/dev/data
/dev/oracle
```

Startup GPFS subsystem

6. To start the GPFS file system, issue the following command:

```
{node1:root}/-> mmstartup -C nodeset1
```

7. Check whether the GPFS subsystem is running (see Example 3-24):

Example 3-24 GPFS subsystem status

```
{node1:root}/-> lssrc -s mmfs
Subsystem         Group            PID          Status
 mmfs             aixmm            1097922      active
```

Allocating disks for GPFS

8. Volume groups and logical volumes used by GPFS are created automatically by the **mmcrlv** command. Specify which physical disks will be used for the GPFS file system in a disk descriptor file.

   The disk descriptor file contains one line for each of the disks to be processed. Disk descriptors have the following format (second and third fields are reserved):

   DiskName:::DiskUsage:FailureGroup

   – Disk Name - Specifies the device name of the physical disk. This can be either an hdisk name or a vpath name for an SDD device. Each disk will be used to create a single volume group and a single logical volume.
   – Disk Usage - Specifies disk usage: dataAndMetadata, dataOnly, metadataOnly. The default value is dataAndMetadata.
   – Failure Group - Specifies a failure group. All disks that have a common point of failure, such as all disks attached to the same adapter, should be placed in the same failure group.

   The disk configuration for our test environment is shown in Example 3-25.

Example 3-25 Physical volume list before running the mmcrlv command

```
{node1:root}/-> lspv
hdisk0          0022be2a28d2eec3                    rootvg          active
hdisk1          none                                None
hdisk2          none                                None
hdisk3          none                                None
hdisk4          0022be2a31bc034d                    None
hdisk5          0022be2a31fa63ca                    None
hdisk6          0022be2a31fa6653                    None
```

```
hdisk7          0022be2a31da8cf8                    None
hdisk8          0022be2a31fa68db                    None
hdisk9          0022be2a31fa6b48                    None
hdisk10         0022be2a470fdfb5                    None
hdisk11         0022be2a31bc034d                    None
hdisk12         0022be2a31fa63ca                    None
hdisk13         0022be2a31fa6653                    None
hdisk14         0022be2a31da8cf8                    None
hdisk15         0022be2a31fa68db                    None
hdisk16         0022be2a31fa6b48                    None
hdisk17         0022be2a470fdfb5                    None
vpath0          none                                None
vpath1          none                                None
vpath2          none                                None
vpath3          none                                None
vpath4          none                                None
vpath5          none                                None
vpath6          none                                None
```

We used GPFS file systems for both Oracle code and data files. Thus, it was better to define a GPFS file system for the Oracle home directory and a *separate* GPFS file system for the Oracle data files.

We wanted to use vpath0 to store the /oracle file system (identified as Oracle Home) and vpath1, vpath2, vpath3, and vpath4 for the /data file system.

For this purpose, we created two disk descriptor files: /gpfs_configure/oracledisk.desc and /gpfs_configure/datadisk.desc, as shown in Example 3-26.

Example 3-26 GPFS descriptor files

```
{node1:root}/-> cat /gpfs_configure/oracledisk.desc
vpath0::::1

(node1:root)#cat /gpfs_configure/datadisk.desc
vpath1::::1
vpath2::::1
vpath3::::1
vpath4::::1
```

9. Issue **mmcrlv** to create the volume groups and logical volumes for the GPFS file systems (see Example 3-27).

Example 3-27 GPFS disk configuration

```
{node1:root}/-> mmcrlv -F /gpfs_configure/oracledisk.desc
{node1:root}/-> mmcrlv -F /gpfs_configure/datadisk.desc
{node1:root}/-> mmlsgpfsdisk

 File system   Disk name    Primary node             Backup node
------------------------------------------------------------------------
 data          gpfs1lv      (directly attached)
 data          gpfs2lv      (directly attached)
 data          gpfs3lv      (directly attached)
 data          gpfs4lv      (directly attached)
 oracle        gpfs0lv      (directly attached)
```

After the **mmcrlv** command, the physical volume and volume group status changes, as shown in Example 3-28.

Example 3-28 Physical volume & Volume group list after running mmcrlv command

```
{node1:root}/-> lspv
hdisk0          0022be2a28d2eec3                    rootvg          active
hdisk1          none                                None
hdisk2          none                                None
hdisk3          none                                None
hdisk4          none                                None
hdisk5          none                                None
hdisk6          none                                None
hdisk7          none                                None
hdisk8          none                                None
hdisk9          0022be2a31fa6b48                    None
hdisk10         0022be2a470fdfb5                    None
hdisk11         none                                None
hdisk12         none                                None
hdisk13         none                                None
hdisk14         none                                None
hdisk15         none                                None
hdisk16         0022be2a31fa6b48                    None
hdisk17         0022be2a470fdfb5                    None
vpath0          0022be2a31bc034d                    gpfs0vg         active
vpath1          0022be2a31fa63ca                    gpfs1vg         active
vpath2          0022be2a31fa6653                    gpfs2vg         active
vpath3          0022be2a31da8cf8                    gpfs3vg         active
vpath4          0022be2a31fa68db                    gpfs4vg         active
vpath5          none                                None
vpath6          none                                None

{node1:root}/-> lsvg
rootvg
gpfs0vg
gpfs1vg
gpfs2vg
gpfs3vg
gpfs4vg
```

The disk descriptor file is rewritten, to be further used as input for the **mmcrfs**, **mmadddisk**, or **mmrpldisk** commands (see Example 3-29).

Example 3-29 GPFS rewritten disk descriptor file

```
{node1:root}/gpfs_configure-> cat oracledisk.desc
gpfs0lv::::1
#MMCRLV_STEP=1
#MMCRLV_STEP=2
{node1:root}/gpfs_configure-> cat datadisk.desc
gpfs1lv::::1
gpfs2lv::::1
gpfs3lv::::1
gpfs4lv::::1
#MMCRLV_STEP=1
#MMCRLV_STEP=2
```

Create the GPFS file system

10. For the Oracle Home file system, since our Oracle9*i* RAC home contains approx. 64000 files, we specified the maximum number of inodes by using the `-N` flag.

    The file systems created will be automatically mounted when the GPFS daemon starts. We created the /oracle and /data GPFS file systems, as shown in Example 3-30.

Example 3-30 Create a GPFS file system

```
# mmcrfs /oracle /dev/oracle -F /gpfs_configure/oracledisk.desc -A yes -B 16K -n 4 -N 80K
# mmcrfs /data /dev/data -F /gpfs_configure/datadisk.desc -A yes -B 16k K -n 4
```

Mount the GPFS file system

11. Mount the GPFS file system on each node as needed:

```
{node1:root}/-> mount /oracle
{node1:root}/-> mount /data
```

For more detail on the GPFS installation and administration, refer to the following documents:

- *General Parallel File System for AIX 5L:AIX Clusters Concepts,Planning,and Installation Guide,* GA22-7895
- *General Parallel File System for AIX 5L:AIX Clusters Administration and Programming Reference,* SA22-7896
- *General Parallel File System for AIX 5L: AIX Clusters Problem Determination Guide,* GA22-7897
- *General Parallel File System for AIX 5L: AIX Clusters Data Management API Guide,* GA22-7898

You can also view the GPFS documentation in HTML format, or you can download the PDF files from:

```
http://www.ibm.com/servers/eserver/pseries/library/gpfs.html
http://www.ibm.com/shop/publications/order (IBM Publications Center)
```

Special considerations for GPFS on SSA disks in an RPD cluster

If you set up a two-node GPFS cluster on an RPD using the IBM 7133 SSA storage system, then in order to use a single-node quorum GPFS configuration, you must configure SSA fencing for the SSA subsystem.

For a two-node GPFS cluster you can choose either a multi-node quorum (which means that if one node is down, the GPFS file system is down, as well), or, by specifying the `-U` option on the **`mmconfig`** command, you can select a single-node quorum configuration.

Single-node quorum allows the remaining node (in a two-node GPFS nodeset) to continue running the GPFS file system in case the peer node fails.

The `node_number` attribute of the ssar router device should be set to the same value as the RSCT peer domain cluster node number. By default, the `node_number` value is 0, which disables SSA fencing.

Use the **`chdev`** command to do this. For example, to set the node_number to a value of 1, type:

```
{node1:root} # chdev -l ssar -a node_number=1
```

If you set up a two-node HACMP/ES cluster, you do not need to enable SSA fencing manually because HACMP enables the SSA fencing when synchronizing the topology.

For more detail about SSA fencing, refer to the following documents:

- *Advanced SerialRAID Adapters User's Guide and Maintenance Information,* SA33-3285
- *Advanced SerialRAID Adapters Technical Reference,* SA33-3286

3.10.3 HACMP 4.5 installation

For our test environment we used High Availability Cluster Multiprocessing Enhanced Scalability version 4.5, since this version has been validated and is supported for Oracle9*i* RAC installations. By the time this book is published, HACMP 5.1 will also be available. Before using this new release of HACMP, check compatibility and any software updates from IBM and Oracle.

With HACMP 5.1, all the different flavors of the previous versions are merged in a single product (no more distinction between HACMP/ES and HACMP/ESCRM). Also, HACMP classic (HAS) has been discontinued in HACMP V5.1.

For more details and the latest updates, see the Oracle certification matrices at:

```
http://otn.oracle.com/support/metalink/content.html
```

HACMP publications

For more details, refer to the official HACMP documentation:

- *High Availability Cluster Multi-Processing for AIX Concepts and Facilities Guide,* SC23-4276
- *High Availability Cluster Multi-Processing for AIX Planning Guide Version 4.5,* SC23-4277
- *High Availability Cluster Multi-Processing for AIX Enhanced Scalability Installation and Administration Guide Version 4.4.1,* SC23-4279
- *High Availability Cluster Multi-Processing for AIX Troubleshooting Guide,* SC23-4280
- *High Availability Cluster Multi-Processing for AIX Programming Locking Applications,* SC23-4281
- *High Availability Cluster Multi-Processing for AIX Programming Client Applications,* SC23-4282
- *High Availability Cluster Multi-Processing for AIX Enhanced Scalability Installation and Administration Guide,* SC23-4284
- *High Availability Cluster Multi-Processing for AIX Master Glossary,* SC23-4285

You can also view or download the documentation in PDF format from:

```
http://www.ibm.com/servers/eserver/pseries/library/hacmp_docs.html
http://www.ibm.com/servers/eserver/pseries/library (IBM pSeries Resource Library)
http://www.ibm.com/shop/publications/order (IBM Publications Center)
```

Product installation

Use either **`smitty installp`** or the installp command line to install the HACMP/ES filesets on all nodes. The Concurrent Resource Manager (CRM) filesets are *not* required, since we are using GPFS for shared storage. The CRM option of HACMP is only for use with RAW devices in a concurrent disk environment.

After the installation, check the fileset levels on all cluster nodes:

```
{node4:root}/-> lslpp -lOu cluster*
```

See Appendix A, “Operating system fileset levels” on page 197 for the fileset versions installed in our environment.

PTF

The following PTF is required. You must have at least the listed level for this fileset:

```
U487607 - cluster.es.server.rte 4.5.0.6
```

Check host equivalence

When synchronizing the cluster definitions on all the nodes, HACMP 4.5 is using “r” commands. Check that the correct host equivalence parameters are set for the root user.

See 3.7.2, “Enabling remote command execution” on page 36 for a description of the remote commands setup.

Post-install environment configuration

To ease further HACMP configuration and operation tasks, we suggest that you add the following directories to your PATH variable:

Example 3-31 HACMP/ES PATH environment

```
PATH=$PATH:/usr/es/sbin/cluster:/usr/es/sbin/cluster/utilities:/usr/es/sbin/cluster/sbin
PATH=$PATH:/usr/es/sbin/cluster/diag:/usr/es/sbin/cluster/etc
export PATH
```

Important: After both RSCT and HACMP have been installed successfully on all the nodes (including latest fixes), all the nodes *must be rebooted* before going on with the HACMP configuration.

3.10.4 HACMP configuration

An HACMP cluster is basically composed of a topology (nodes, networks) and resources (service adapters for IP address takeover (IPAT), disks, and scripts).

For Oracle9*i* RAC, we only need to configure the cluster topology. No resource configuration is needed, since we are not using raw devices but GPFS. The volume groups are defined in the GPFS configuration. Also, no IP address takeover (IPAT) is configured. Remember that Oracle interconnect networks do not support IPAT.

The cluster configuration is defined on one node. This definition is then synchronized to the rest of the nodes. This feature is called Cluster Single Point Of Control (CSPOC). However, when you are creating or modifying the cluster definition, make sure that there is no other similar operation in process. After each configuration change, the cluster topology must be synchronized.

Cluster configuration steps:

- Define the cluster.
- Define the cluster nodes.
- Define the IP networks (which will be used for the Oracle cache fusion traffic).
- Define the serial networks (if applicable) used for avoiding “split brain” situations.

Figure 3-10 on page 61 displays the SMIT menu tree, with all the steps needed to configure the nodes and the interconnect networks.

Figure 3-11 on page 62 shows how to configure the non-IP serial network for the HACMP heartbeat.

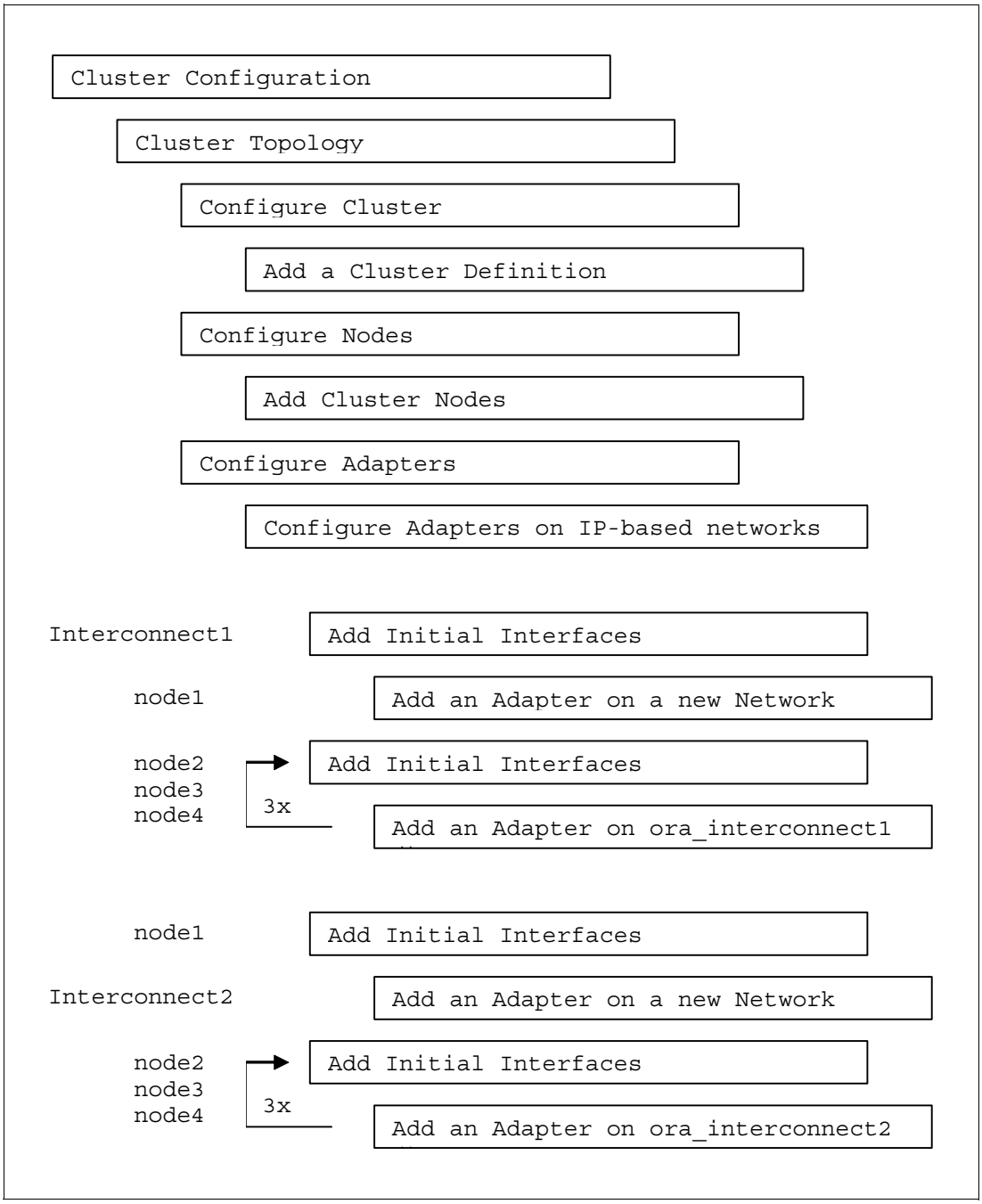


Figure 3-10 Smit menus flowchart for cluster node and interconnect network configuration

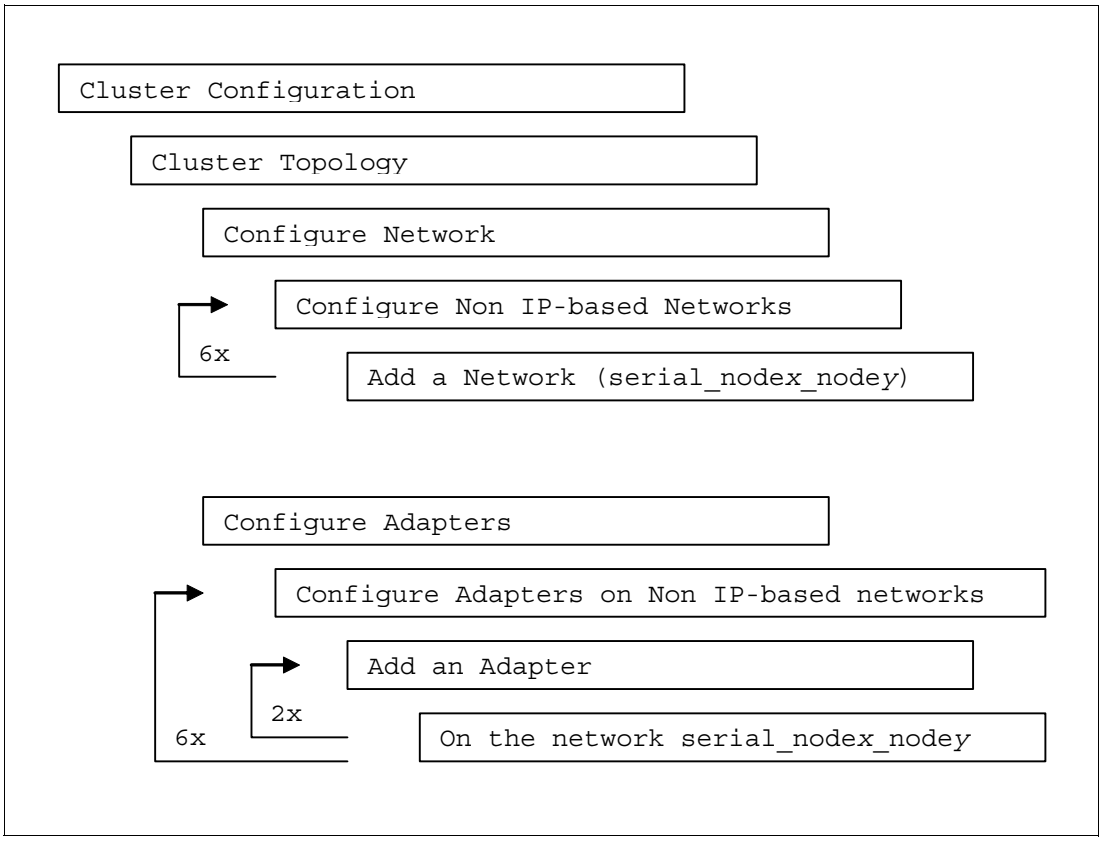


Figure 3-11 Smit menus flowchart for non-IP serial network configuration

3.10.5 HACMP cluster and nodes definition

This step defines the cluster, and the nodes composing the cluster.

To get started, enter **`smitty hacmp`**, and follow the sequence shown in the following figures.

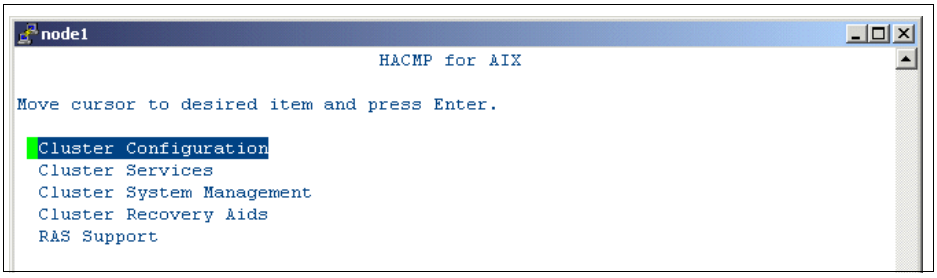


Figure 3-12 HACMP entry screen

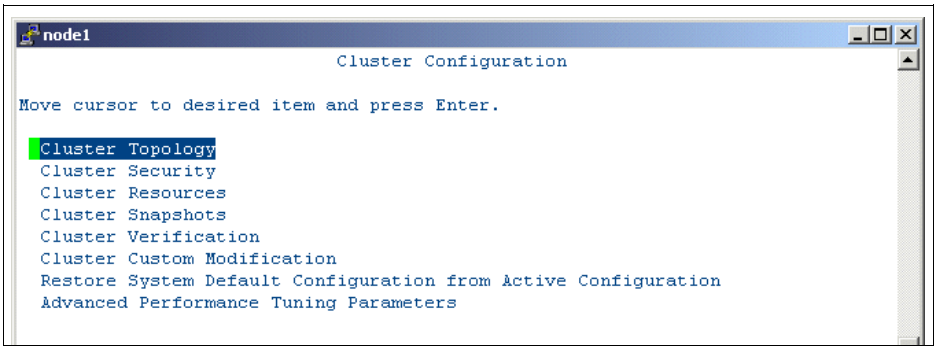


Figure 3-13 HACMP topology main screen

The cluster topology consists of:

- Cluster name
- Cluster nodes
- Cluster networks (IP.Serial ATM)
- Adapters connected to these networks

The only networks configured in HACMP for our environment are the Oracle interconnects and the serial links for heart beat.

If you want a second failover level for the Oracle9*i* RAC cache fusion traffic, the client network can also be defined as “public” in the HACMP configuration.

The GPFS cluster is operating in our environment on top of an RSCT Peer Domain cluster. Thus, the network assigned to GPFS is not defined inthe HACMP configuration.

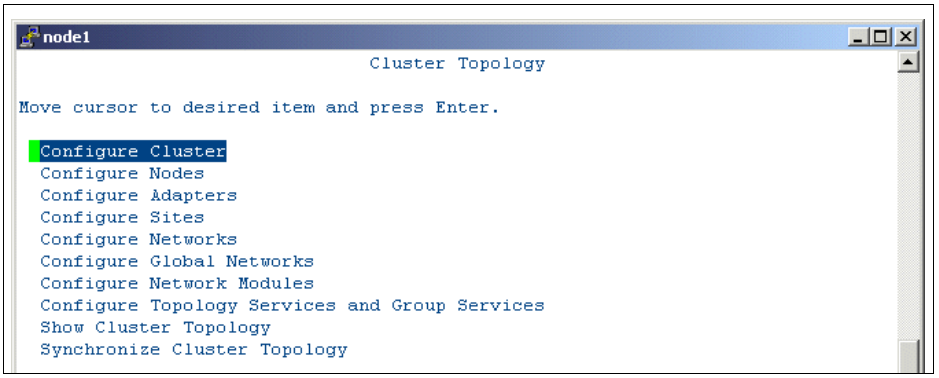


Figure 3-14 HACMP cluster primary configuration

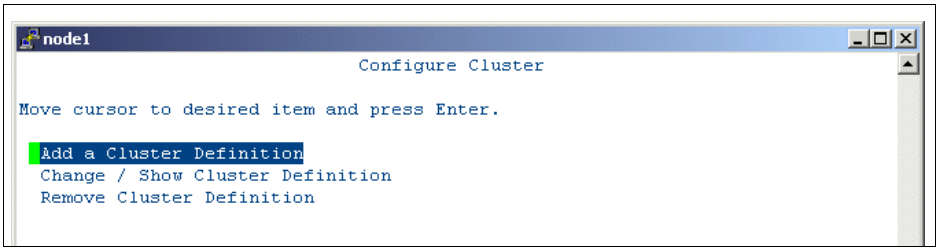


Figure 3-15 Add a cluster definition

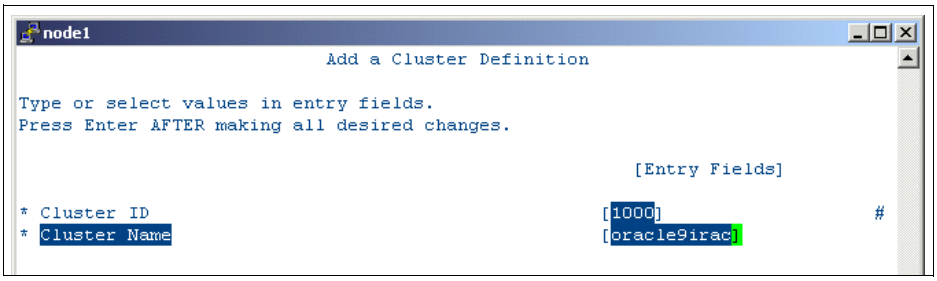


Figure 3-16 Naming and identifying the HACMP cluster

The cluster ID is a unique number. The cluster name is a label of your choice.

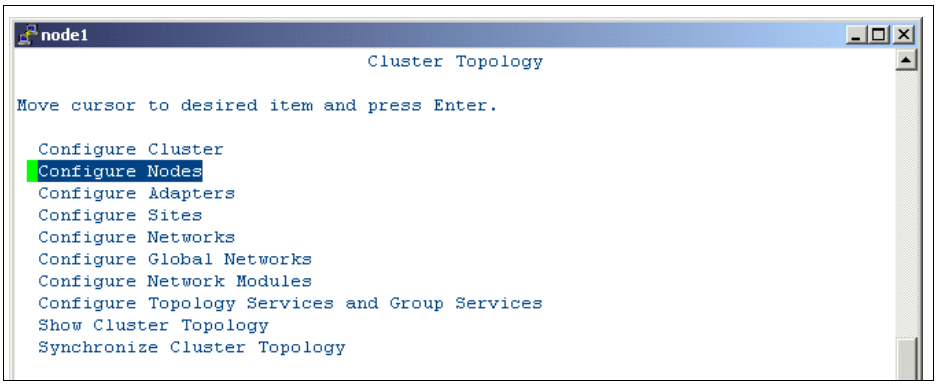


Figure 3-17 Configure the nodes

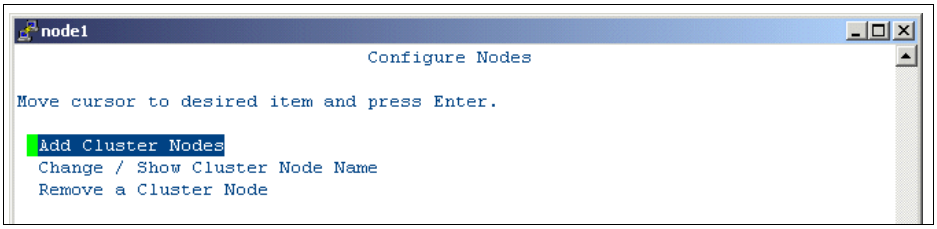


Figure 3-18 Add cluster nodes

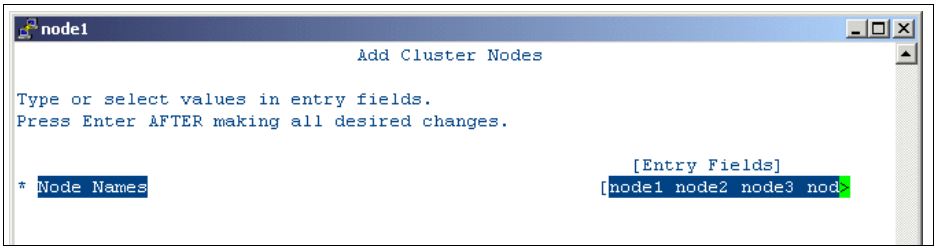


Figure 3-19 Entering the cluster nodes

You can enter any node name of your choice. For clarity, we chose to use the hostnames (identified by the **hostname** command and associated with the IP label of the first network adapter, - en0).

You can define multiple nodes at once, separated by spaces.

3.10.6 HACMP IP networks

This step defines the IP networks HACMP manages.

Which IP networks should be defined in the HACMP configuration?

The networks that are part of the HACMP cluster are only a subset of the networks we use in our test environment. Only the networks intended for Oracle9*i* RAC Interconnect, and the serial networks for HACMP non-IP heart beat are defined in this HACMP cluster.

If GPFS is defined in an RPD cluster, the GPFS network should not be defined into HACMP.

For a second level of failover purpose, a public network could be defined into the HACMP cluster (for example, the client and administrative network). Defining this network into HACMP will allow Oracle to use it in case the other networks, defined as “private”, fail.

However, we do not recommend this configuration because of some Oracle constraints. For more details about this issue, including the client network configuration, see “How HACMP networks are used by Oracle” on page 78.

There are two reasons why Oracle 9i RAC prefers a “private” to a “public” service network:

- For performance reasons: Oracle cache fusion traffic should be separated from any client communication to prevent communication bottlenecks.
- A private network (physically isolated from any outside IP traffic) ensures data security (since Oracle uses UDP for interconnect).

A typical HACMP network configuration in an Oracle9*i* RAC environment consists of:

- One primary interconnect network (labeled “private” in HACMP)

  We recommend a GBit ethernet for this network, or an SP Switch.

  In normal operation, only this primary “private” network is used for interconnect traffic.
- One secondary interconnect private network

  We recommend at least a GBit ethernet. This interconnect is for backup only, not used by Oracle in normal operation.
- A set of serial networks for HACMP non-IP heart beat

  The number of networks depends on the number of nodes (one for a 2-node cluster, six for a 6-node cluster).

Defining the primary interconnect network

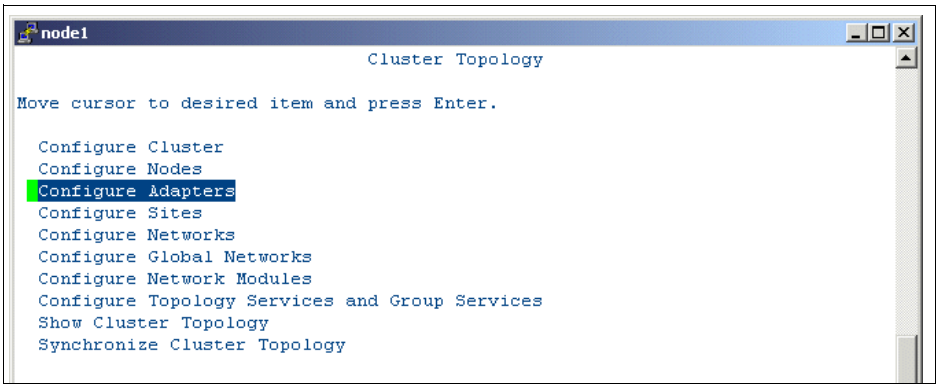


Figure 3-20 Configuring the adapters

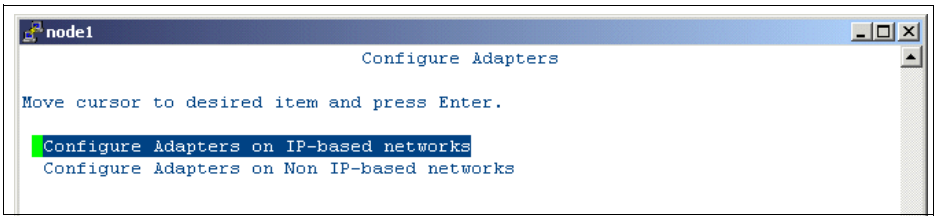


Figure 3-21 Configuring the IP-based adapters

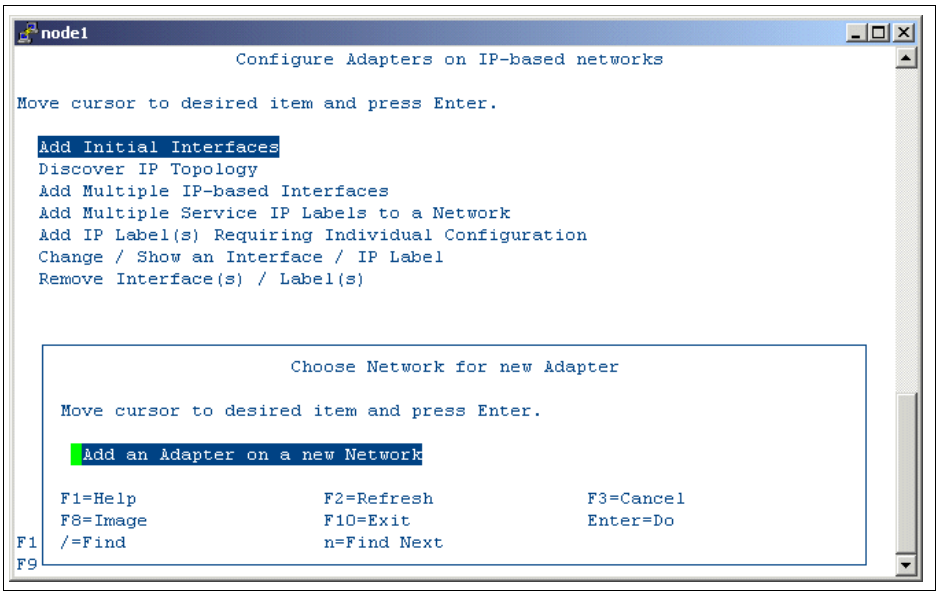


Figure 3-22 Adding the first interconnect

If you chose “Add an adapter on a new Network”, the network object will be created first. You have to specify the name on the next screen.

Keep in mind that you can also first define the network object, with no adapters, then define the adapters (IP labels) contained in this network.

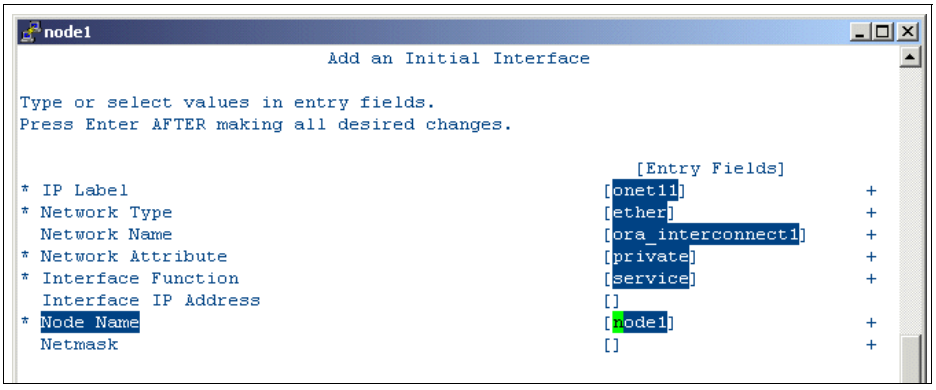


Figure 3-23 Adding the interconnect1 interfaces

Be sure to define the network characteristics and the name of the interface (with the associated IP label) connected to this network for each node. Add the adapters corresponding to all four cluster nodes.

Important: Be sure to set the network attribute to “private”. When selecting interconnect networks, Oracle uses the ones labeled as private.

Defining the secondary interconnect

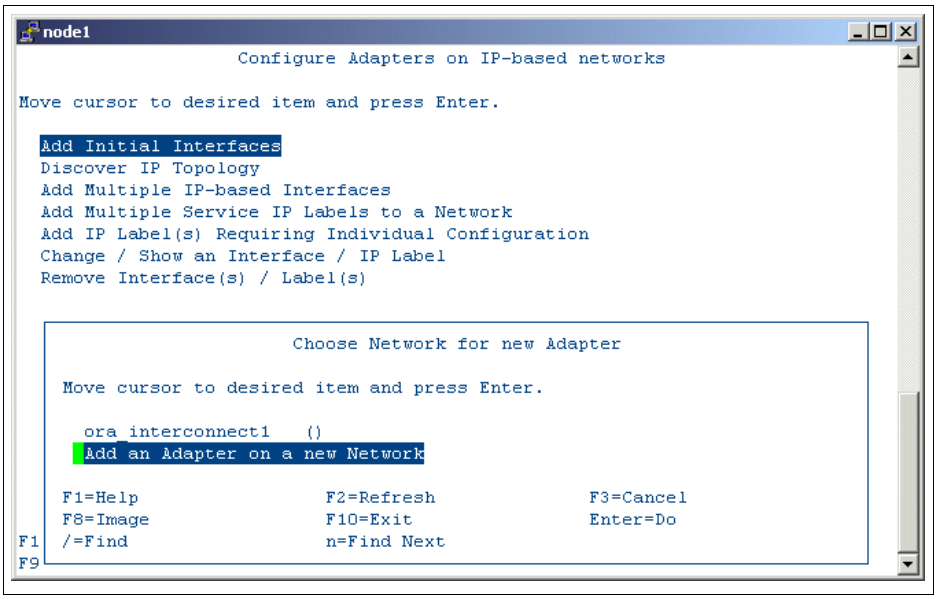


Figure 3-24 Adding the second interconnect

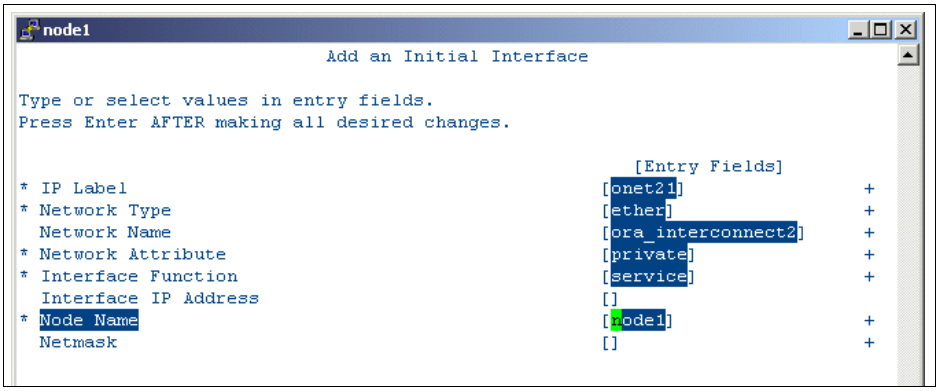


Figure 3-25 Adding the interconnect2 interfaces

Use the same procedure to configure the second interconnect.

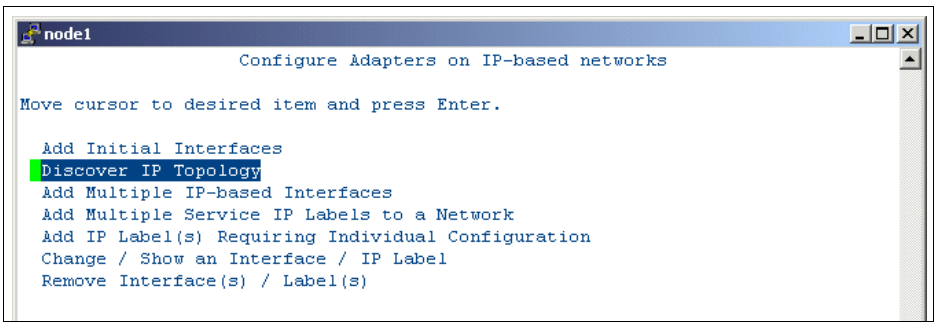


Figure 3-26 Discovering the topology

Although HACMP 4.5 provides an automatic configuration discovery feature, we suggest that you use manual definition to avoid confusion caused by automatic naming of the network objects.

Verifying cluster definition and topology synchronization

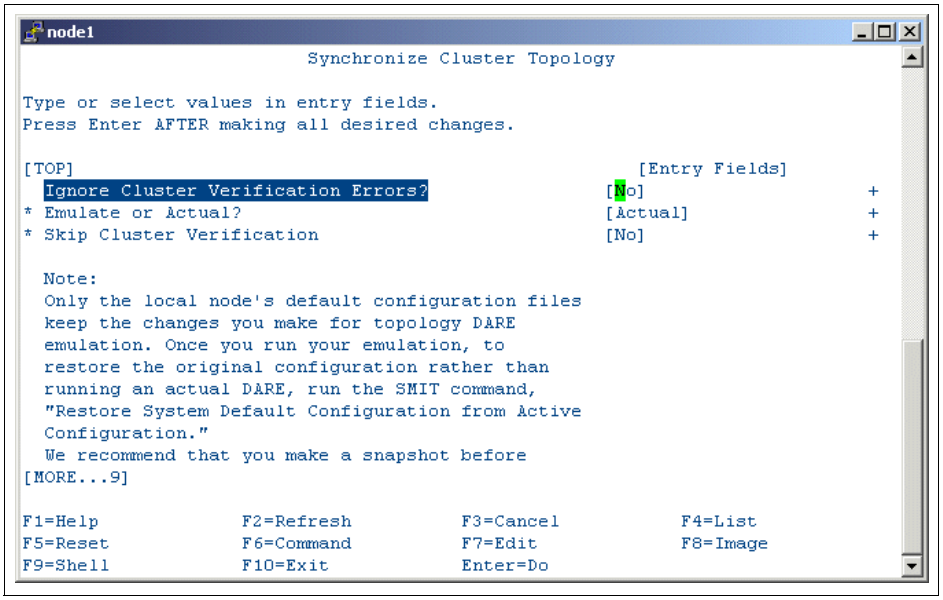


Figure 3-27 Synchronization process

For this operation (see Figure 3-27) the first step is to check the actual cluster configuration. Then, if everything is correct, this configuration is sent to the other nodes defined in the cluster (synchronization). It is mandatory to synchronize after any configuration change. HACMP will not start if it detects differences in the cluster configuration of the nodes. If the synchronization fails, correct the problem, and synchronize again.

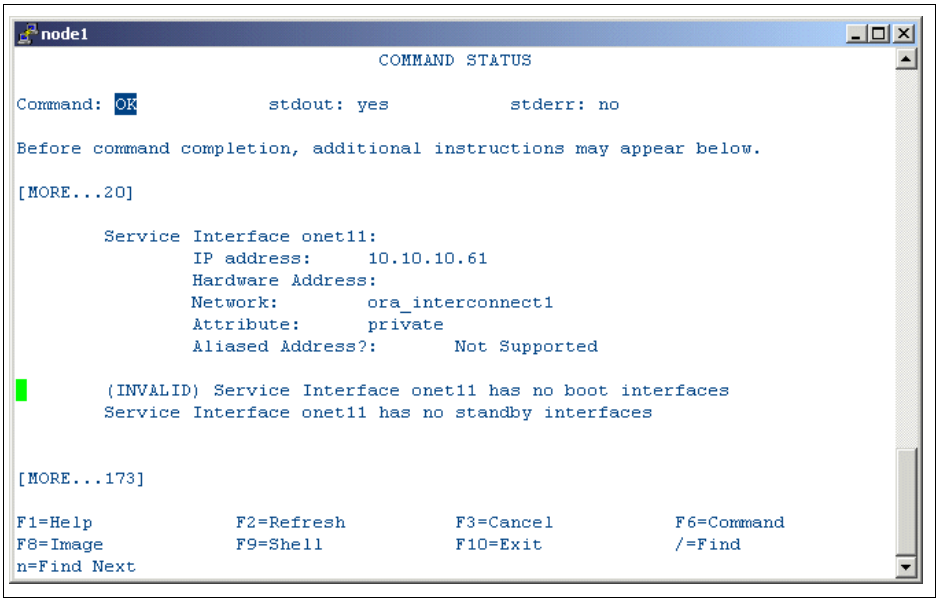


Figure 3-28 Synchronization succeeded

Check for “Warning” labels, even if cluster synchronization is “OK”. Since we do not use IP takeover, these warning messages are normal.

3.10.7 HACMP serial networks

This step defines the non-IP networks by specifying the tty devices HACMP should use for the non-IP heart beat.

Adding the non-IP serial networks for the HACMP heart beat

To differentiate between IP subsystem failure, node failure, and network failure, HACMP needs a non-IP network. This network carries the non-IP heart beat traffic. If a problem occurs on a network interface (used for IP traffic), or the TCP/IP stack fails, HACMP can differentiate a network failure from a node failure, using the non-IP network. This link helps HACMP in handling the “split brain” situations. This serial network is not used by Oracle.

The most common non-IP network uses the RS232 serial interfaces. Other serial network choices are target mode SSA (tmssa) and target mode SCSI (tmscsi). All these are point-to-point, non-IP links.

For a 2-node cluster, a single serial link is enough. For a 4-node cluster, we need six point-to-point links to enable direct node-to-node communication. Each node must be linked to all the others. Consider using multiport serial adapters for a configuration consisting of more than two nodes and null-modem RS232 serial cables.

Figure 3-29 shows the serial network topology based on a 4-node cluster.

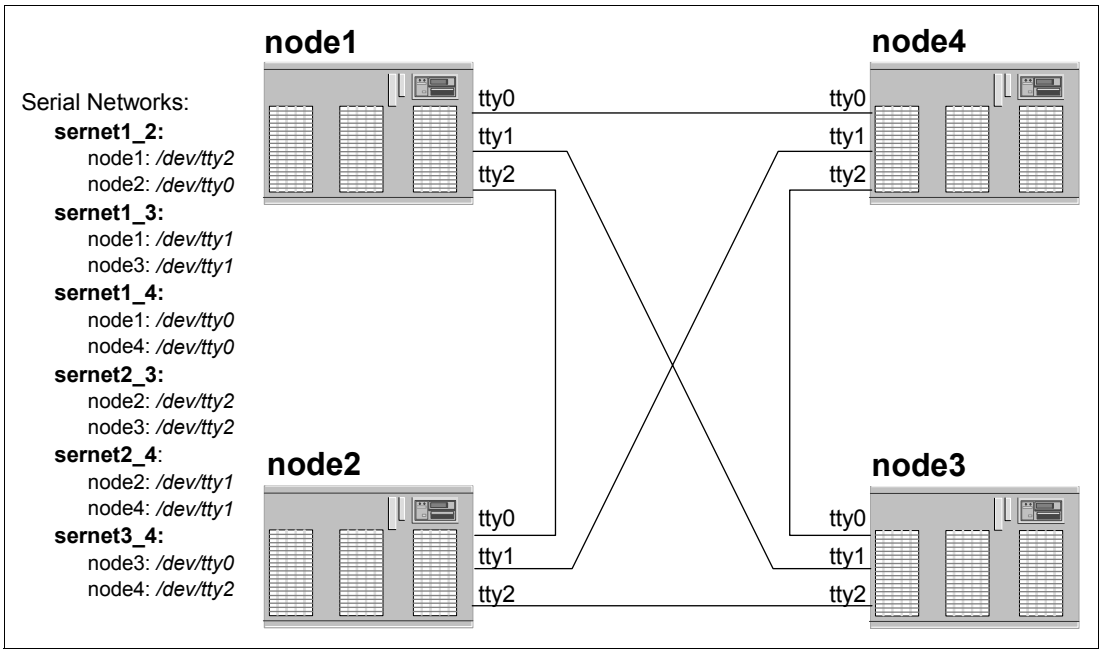


Figure 3-29 Serial network topology

Configuring the tty devices on the nodes

For serial network configuration, the first step is to create the serial communication devices.

Since we have four nodes, we need six serial networks. Thus we have to define the necessary number of tty devices (in our case, we need three tty devices, as shown in Figure 3-29).

Using **smitty tty**, follow the steps shown in the following screens.

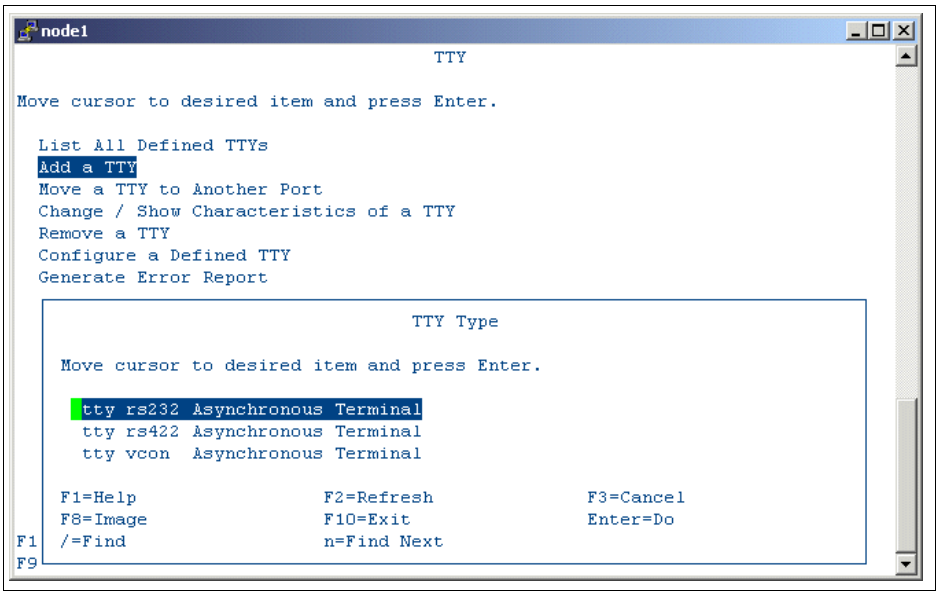


Figure 3-30 Add a tty

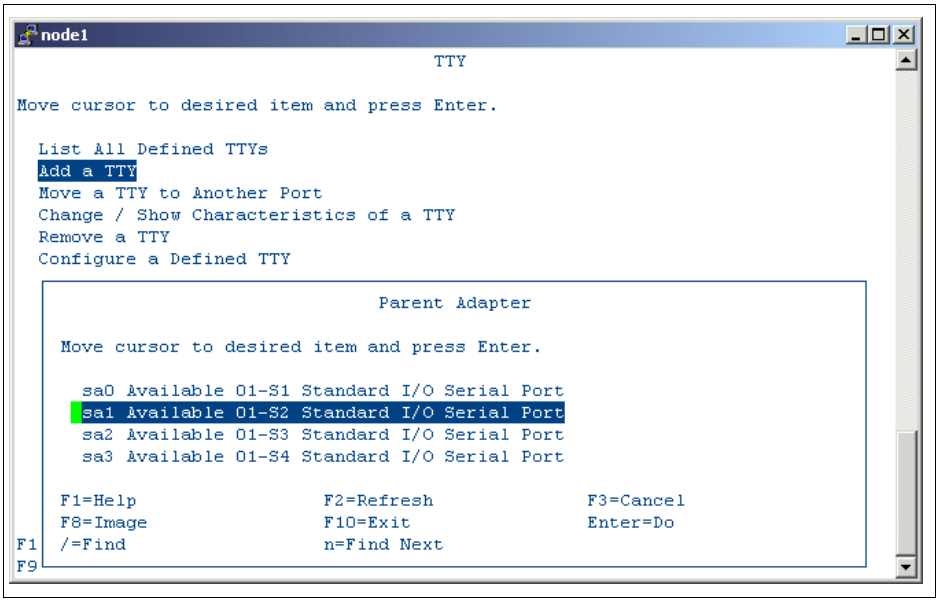


Figure 3-31 Add a tty on a port

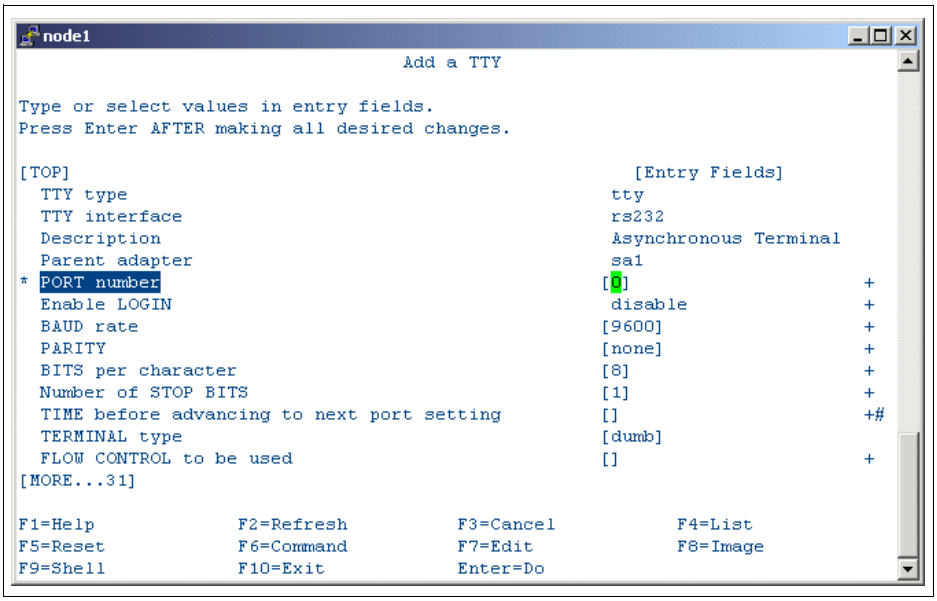


Figure 3-32 Configuring a new tty device

Using the same steps, configure all needed tty devices.

You should have the following ttys on all the nodes, as shown in Example 3-32.

Example 3-32 tty devices used for serial networks

```
{node1:root}/dev-> ls -l /dev/tty?
crw-------   1 root     system       17,  0 Jun 10 15:42 /dev/tty0
crw-rw-rw-   1 root     system       17,  1 Jun 10 16:06 /dev/tty1
crw-rw-rw-   1 root     system       17,  2 Jun 10 16:50 /dev/tty2
```

Configuring HACMP serial networks

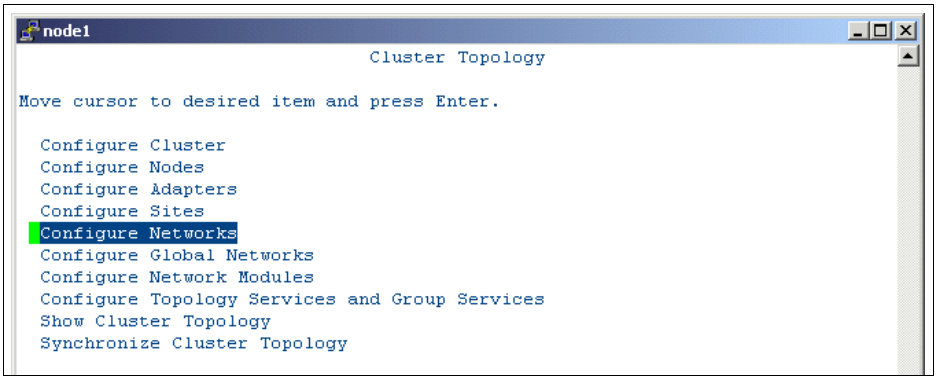


Figure 3-33 Configuring the serial network

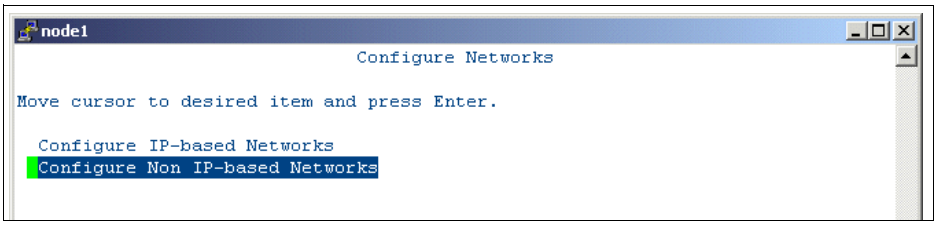


Figure 3-34 Serial non-IP network

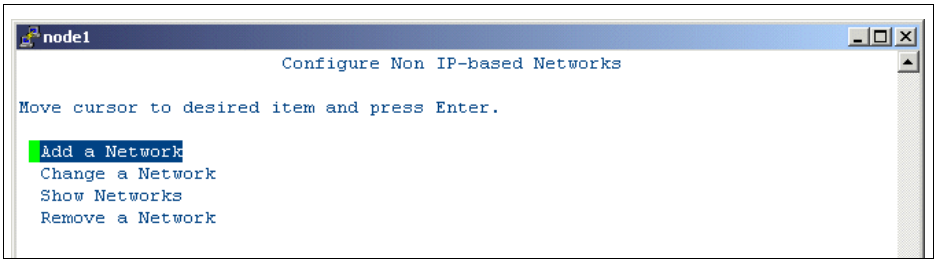


Figure 3-35 Adding a serial network

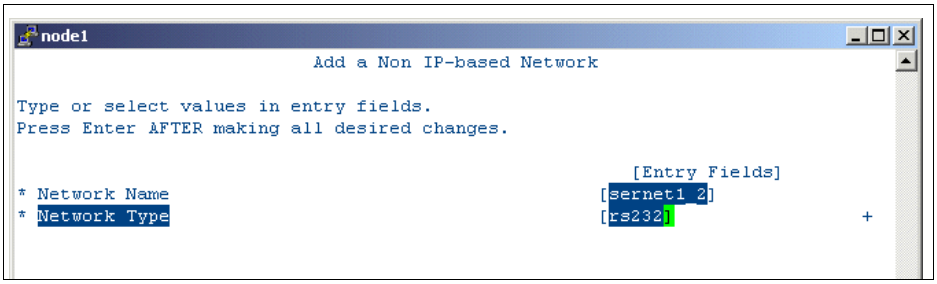


Figure 3-36 Entering the characteristics of the serial network

Repeat this screen to create all the required serial networks.

Configuring the HACMP serial adapters

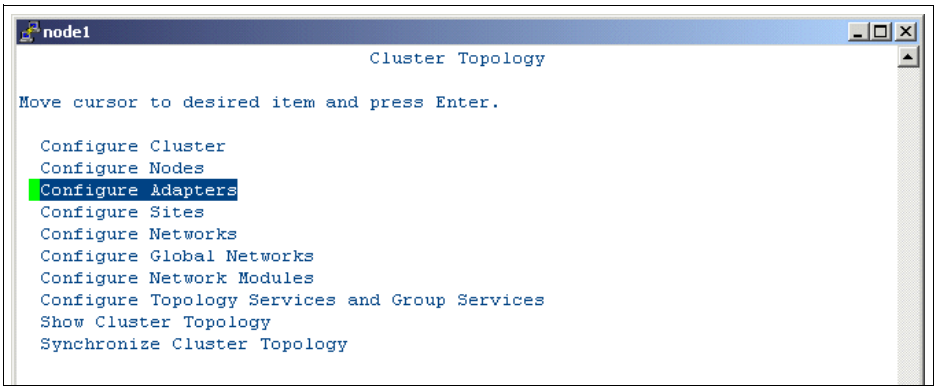


Figure 3-37 Configuring the serial adapters

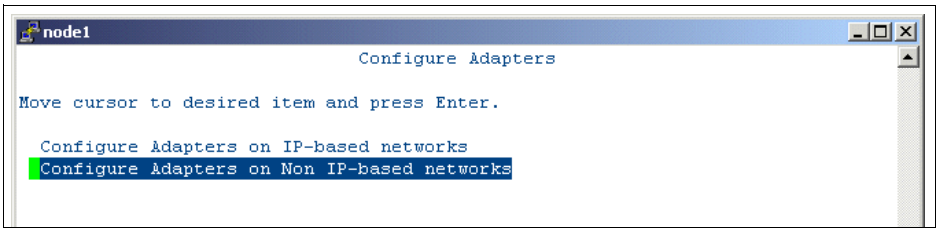


Figure 3-38 Configuring the non-IP serial adapters

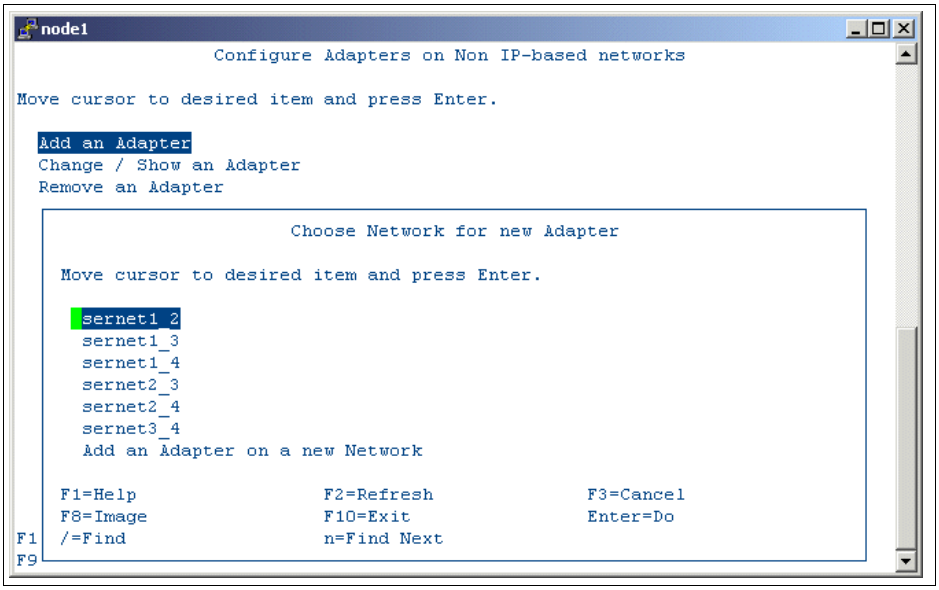


Figure 3-39 Adding a serial adapter

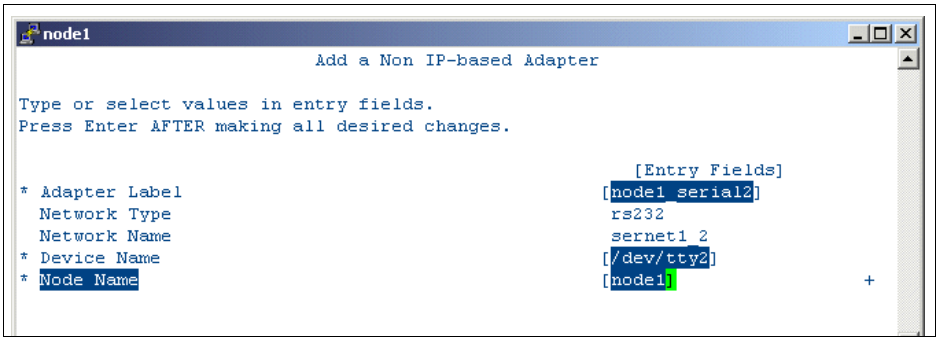


Figure 3-40 Entering the characteristics of a serial adapter

Each serial network contains only two devices/adapters. Use suggestive names for the serial networks, and double-check the configuration before proceeding to cluster synchronization.

Verify the cluster definition and synchronization

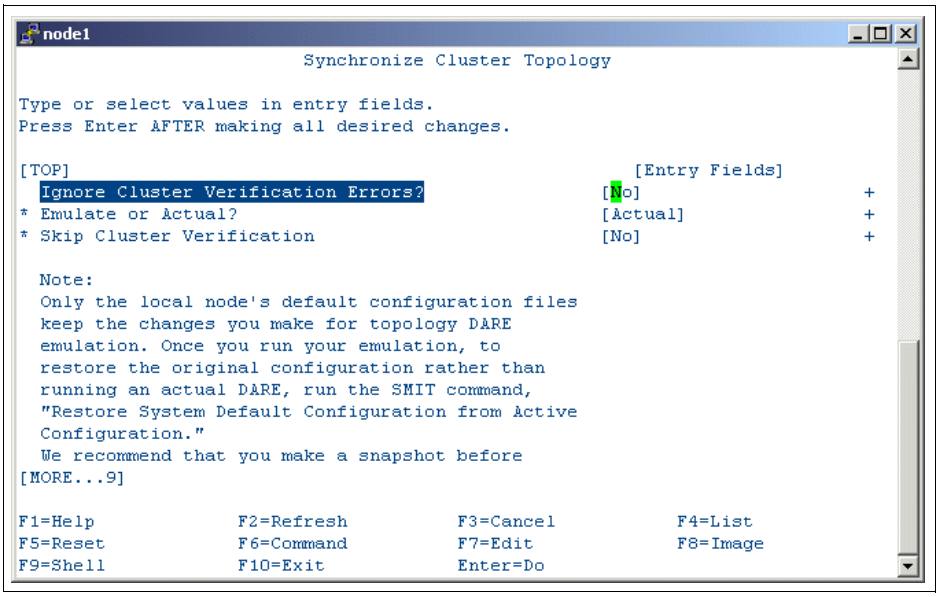


Figure 3-41 Last synchronization

This completes the cluster configuration for our Oracle9*i* RAC environment.

3.10.8 HACMP configuration considerations

To be sure that your HACMP cluster definition is correct, check the output of the **cllsif** command. If you don't have the PATH variable correctly set up, use:

```
{root@node1}/_> /usr/es/sbin/cluster/utilities/cllsif
```

Note: You should not use any kind of IP address takeover in a cluster designed for Oracle9*i* RAC. You should avoid defining *any* resource groups in the HACMP cluster. The disk definition is part of the GPFS configuration.

See Example 3-33 for a typical interconnect configuration. Use the **cllsif** command to verify the HACMP network configuration.

Example 3-33 HACMP cllsif command output for private interconnect networks

```
{node1:root}/-> /usr/es/sbin/cluster/utilities/cllsif |grep interconnect

Adapter  Type       Network           Net Type   Attribute  Node   IP Address     Interface

onet11   service    ora_interconnect1 ether      private    node1  10.10.10.61    en1
onet21   service    ora_interconnect2 ether      private    node1  172.16.10.61   en2

onet12   service    ora_interconnect1 ether      private    node2  10.10.10.62    en1
onet22   service    ora_interconnect2 ether      private    node2  172.16.10.62   en2

onet13   service    ora_interconnect1 ether      private    node3  10.10.10.63    en1
onet23   service    ora_interconnect2 ether      private    node3  172.16.10.63   en2

onet14   service    ora_interconnect1 ether      private    node4  10.10.10.64    en1
onet24   service    ora_interconnect2 ether      private    node4  172.16.10.64   en2
```

Example 3-34 shows the serial networks for a 4-node cluster.

Example 3-34 HACMP cllsif command output for serial networks

```
{node1:root}/-> /usr/es/sbin/cluster/utilities/cllsif |grep rs232

Adapter             Type       Network          Net Type   Attribute  Node   IP Address

node1_serial0       service    sernet1_4        rs232      serial     node1  /dev/tty0
node1_serial1       service    sernet1_3        rs232      serial     node1  /dev/tty1
node1_serial2       service    sernet1_2        rs232      serial     node1  /dev/tty2

node2_serial0       service    sernet1_2        rs232      serial     node2  /dev/tty0
node2_serial1       service    sernet2_4        rs232      serial     node2  /dev/tty1
node2_serial2       service    sernet2_3        rs232      serial     node2  /dev/tty2

node3_serial0       service    sernet3_4        rs232      serial     node3  /dev/tty0
node3_serial1       service    sernet1_3        rs232      serial     node3  /dev/tty3
node3_serial2       service    sernet2_3        rs232      serial     node3  /dev/tty2

node4_serial0       service    sernet1_4        rs232      serial     node4  /dev/tty0
node4_serial1       service    sernet2_4        rs232      serial     node4  /dev/tty1
node4_serial2       service    sernet3_4        rs232      serial     node4  /dev/tty2
```

See also Appendix C, “HACMP cluster configuration output” on page 221 for the cluster definition used in our test environment.

3.10.9 HACMP start/stop and monitoring

Starting HACMP

Once HACMP is configured and synchronized, the cluster can be started. The startup can be done in two different ways:

- Starting HACMP on each node separately

  Log on as `root` on each node, and enter **`smitty clstart`**.

  Be sure to also start the Startup Cluster Information Daemon. This enables you to run the **`clstat`** command, which displays the current state of the cluster. During the HACMP startup process on the first node, repeat these steps on the other nodes.

  Even if the result from the SMIT screen shows “OK”, be sure to double-check the HACMP processes after the necessary stabilization period. Stabilization may take between 30 seconds and 5 minutes, depending on the cluster configuration.

- Starting HACMP services on all nodes from a single node

  On a single node, enter **`smit hacmp`**, and select the menus shown in Figure 3-42.

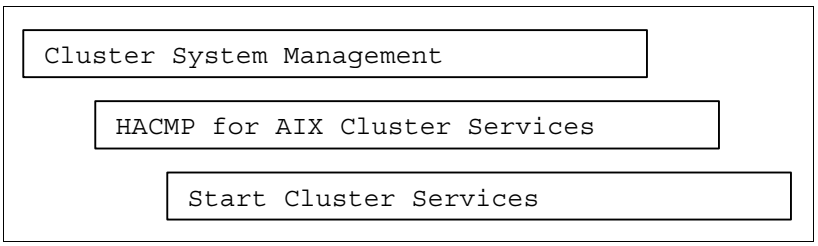


Figure 3-42 SMIT path to automatically start HACMP on all the nodes

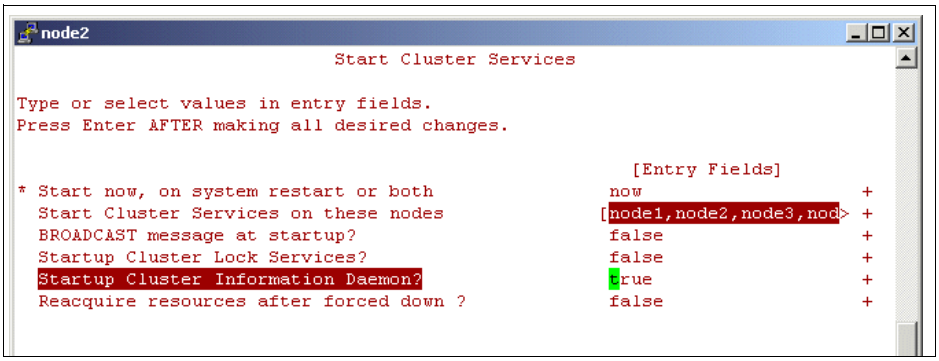


Figure 3-43 HACMP Cluster Single Point Of Control (CSPOC) startup

Also start the Startup Cluster Information Daemon. This daemon (clinfoES) is required to run the **clstat** command, which displays the current state of the cluster.

This process is longer than the first one, because it waits for HACMP services to stabilize on the first node before starting them on the second node, and so on.

Stopping HACMP

Important: Before stopping HACMP, be sure that all the Oracle9*i* RAC instances are stopped.

As with starting HACMP, there are two methods for stopping it. Use the corresponding set of SMIT screens to stop HACMP.

When prompted to choose the shutdown mode:

- "Graceful" is recommended.
- Use "forced" only if "graceful" does not stop the cluster. This is equivalent to **kill -9**.

Even with the forced option, it takes several minutes for HACMP to stop completely. Be sure to check the cluster log files and any cluster-related processes before starting again. Cluster recovery may be needed if the "forced" option was used to stop the cluster services.

Useful commands

- **clstat** or **xclstat**

  These commands need the clinfoES daemon to run on the nodes to work properly. Refer to "Starting HACMP" on page 75 for more details.

  clstat -a is the ASCII display version and can be used on any terminal (see Figure 3-44 on page 77). **xclstat** is a graphical version, and requires an X11 environment (see Figure 3-45 on page 77).

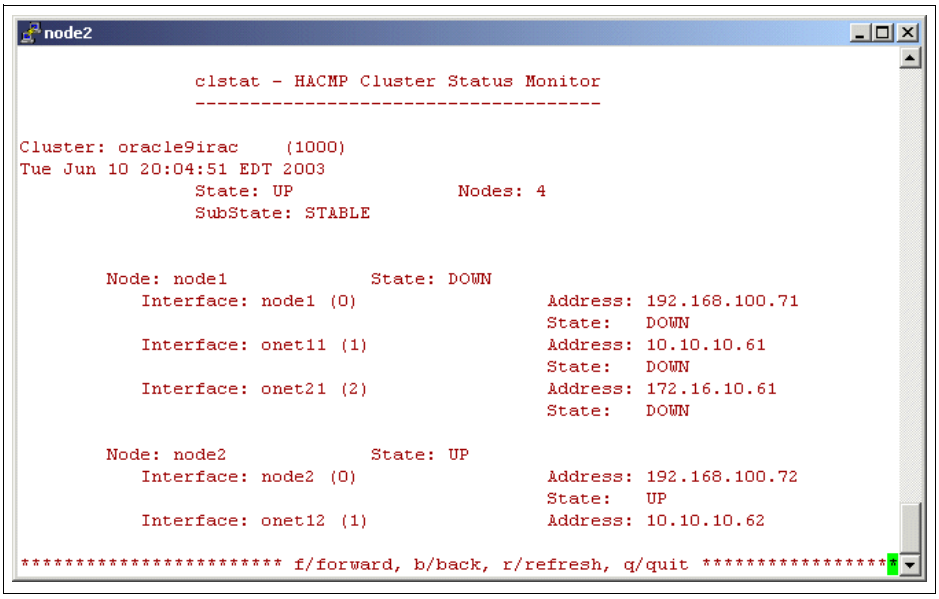


Figure 3-44 clstat -a command output

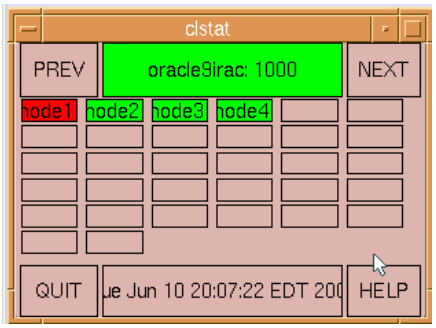


Figure 3-45 xclstat command output

- **`lssrc -a | grep -E "ES|svcs"`** lists the state of the HACMP subsystems.

  In Example 3-35, we can see that HACMP is up and running. The Group Services GLobalized Switch Membership (grpglsm), which is used to provide global synchronization in an SP Switch environment, is not used for our platform.

Example 3-35 lssrc -a | grep -E "ES|svcs"

```
{node2:root}/-> lssrc -a | grep -E "ES|svcs"
 topsvcs          topsvcs          1253488      active
 grpsvcs          grpsvcs          655364       active
 emsvcs           emsvcs           848026       active
 emaixos          emsvcs           508030       active
 clstrmgrES       cluster          843836       active
 clsmuxpdES       cluster          766166       active
 clinfoES         cluster          1306780      active
 grpglsm          grpsvcs                       inoperative
```

- **`clverify`** is an interactive program that performs HACMP installation and configuration verification.
- **`cllsif`** and **`cllscf`** display the cluster configuration. For more information about these commands, refer to Appendix C, “HACMP cluster configuration output” on page 221.

3.10.10 HACMP in an Oracle9*i* RAC environment

Adding a "public" network to the HACMP cluster

In case of a primary interconnect failure, Oracle9*i* RAC fails over to the secondary interconnect. You can add an extra level of failover, in case the two "private" interconnects fail at the same time. The network used for client and administration operations can be configured in the HACMP cluster definition, and marked as "public".

We recommend that you do *not* include this public network in any resource group in the cluster configuration. We tested this failure scenario, and we saw that if both private interconnects fail, Oracle9*i* RAC fails over to the public network.

Athough, in this case, the database is operating in degraded mode because the client network may be too slow to sustain both interconnect and client traffic, the Oracle9*i* RAC cluster will continue to run.

How HACMP networks are used by Oracle

All the networks used by Oracle9*i* RAC are only defined in the HACMP cluster topology. There is no other configuration file defining these networks.

- Networks can be configured into a private or public category. Oracle chooses the private networks first, which are preferred over public networks.
- If more than one private (or public) network is chosen, then they are ordered by performance quality:
  a. High Performance Switch (in an SP environment)
  b. FDDI
  c. Ethernet
- Oracle chooses only networks configured as "service," and can choose up to three networks. If more than one network is selected, Transparent Network Failover Failback (TNFF) is enabled. At any point in time, interconnect IPC traffic is sent over only one of the networks available, even though multiple networks can be chosen. In this configuration we have not observed any load balancing; multiple networks are used for high availability purposes.
- If the parameter cluster_interconnects is set in the init.ora file, TNFF is disabled. Even when setting this parameter, it is important to use "private" networks as well.

Examples

- If a cluster has three private Ethernet networks and one public high-performance switch (HPS) network, Oracle chooses the three Ethernet networks for its IPC.
- In the case where a cluster has one private FDDI network, two public Ethernet networks, and one public HPS network, Oracle chooses in the following order:
  a. Private FDDI network (first preferred network for IPC)
  b. Public HPS network (second preferred network for IPC)
  c. Public Ethernet first returned by the **`cllsif`** command (third preferred network for IPC)
- If two Ethernet private networks are defined in HACMP, Oracle will choose as primary interconnect the one that is returned first by the **`cllsif`** command, so be very careful when choosing network names.

In the **`cllsif`** command output shown in Example 3-36 on page 79, Oracle uses the networks in this order:

1. ora_interconnect1 (first private network found in the **`cllsif`** output)

2. ora_interconnect2 (second private network found in the **cllsif** output)
3. client_network (first public network found in the **cllsif** output)

Example 3-36 cllsif ordered networks list

```
{node1:root}/-> cllsif
Adapter  Type       Network           Net Type   Attribute  Node   IP Address     Interface

node1    service    client_network    ether      public     node1  192.168.100.71 en0
onet11   service    ora_interconnect1 ether      private    node1  10.10.10.61    en1
onet21   service    ora_interconnect2 ether      private    node1  172.16.10.61   en2
node2    service    client_network    ether      public     node2  192.168.100.72 en0
onet12   service    ora_interconnect1 ether      private    node2  10.10.10.62    en1
onet22   service    ora_interconnect2 ether      private    node2  172.16.10.62   en2
node3    service    client_network    ether      public     node3  192.168.100.73 en0
onet13   service    ora_interconnect1 ether      private    node3  10.10.10.63    en1
onet23   service    ora_interconnect2 ether      private    node3  172.16.10.63   en2
node4    service    client_network    ether      public     node4  192.168.100.74 en0
onet14   service    ora_interconnect1 ether      private    node4  10.10.10.64    en1
onet24   service    ora_interconnect2 ether      private    node4  172.16.10.64   en2
```

Logs

The HACMP main log file is /tmp/hacmp.out.

Although this file is duplicated on each node, its contents on each node may differ, depending on cluster status and event succession. The HACMP startup process may take several minutes. To check events succession, display the content of this file on each node, as shown in Example 3-37.

Example 3-37 /tmp/hacmp.out log file

```
{node2:root}/-> tail -f /tmp/hacmp.out
                        HACMP Event Summary
Event: /usr/es/sbin/cluster/events/check_for_site_up_complete node1
Start time: Tue Jun 10 20:28:50 2003

End time: Tue Jun 10 20:28:50 2003

Action:         Resource:                       Script Name:
------------------------------------------------------------------------------
No resources changed as a result of this event
------------------------------------------------------------------------------
```

In case of problems, another log that can be checked is /tmp/clstrmgr.debug.

3.11 Check list

For a quick overview of the platform installation and configuration process, we offer this check list. Unless otherwise specified, the operations in this list should be performed on all cluster nodes.

1. RAM > 512 MB (**lsattr -El sys0 -a realmem**)

   Refer to 3.6.5, “Memory requirements” on page 32.

2. 64-bit hardware processors (**getconf HARDWARE_BITMODE**)

   Refer to 3.6.3, “AIX 5L 32/64-bit kernel considerations” on page 31.

3. maxuproc > 2000 (**lsattr -El sys0 -a maxuproc**)

   Refer to 3.6.8, “Environment and user settings” on page 34.

4. Oracle user unlimited (logged as Oracle user, **ulimit -a**)

   Refer to 3.6.8, “Environment and user settings” on page 34.

5. Root user nofiles(descriptors) = 2000, and not -1 (**cat /etc/security/limits**)

   Refer to 3.6.8, “Environment and user settings” on page 34.

6. PTF U487607 - cluster.es.server.rte >= 4.5.0.6 (**lslpp -l cluster.es.server.rte**)

   This fileset must be to this level, or higher.

   Refer to “HACMP 4.5 filesets” on page 207.

7. PTF U486402 - mmfs.base.cmds >= 3.5.0.2 (**lslpp -l mmfs.base.cmds**)

   This fileset must be to this level, or higher.

   Refer to “GPFS 2.1 filesets” on page 202.

8. AIX 5.2 ML1 (**oslevel -r**)

   Refer to “AIX 5.2 ML1 base operating system filesets” on page 198.

9. Level of RSCT filesets (**lslpp -l rsct***)

   Refer to “RSCT 2.3.1 filesets” on page 201.

10. Level of GPFS filesets (**lslpp -l mmfs***)

    Refer to “GPFS 2.1 filesets” on page 202.

11. Level of HACMP filesets (**lslpp -l cluster***)

    Refer to “GPFS 2.1 filesets” on page 202.

12. Level of ESS 2105 and SDD filesets (**lslpp -l ibm***)

    Are you using the proper version of SDD? Refer to 3.9.5, “ESS Subsystem Device Driver setup” on page 48.

    Refer to “Enterprise Storage Server (ESS) filesets” on page 208.

13. FC adapter microcode level >= 02C03891 (or 3.82A1) (**lscfg -vl fcs0**)

    Refer to 3.9, “Cluster nodes SAN configuration” on page 45.

14. At least 800 MB of free space in the /tmp, or temporary file system chosen for the Oracle Universal Installer (**df -k /tmp**)

    Refer to 3.6.7, “Temporary space” on page 33.

15. hagsuser group created before executing **rootpre.sh** (**cat /etc/group**)

    If you are unsure, rerun the Oracle script **rootpre.sh** located on the first Oracle CD-ROM.

    Refer to “Defining an Oracle administrative user and group” on page 87.

16. Network parameters (**no -a | grep udp**)

    ```
    udp_sendspace=65536
    udp_recvspace=655360
    ```

    Refer to “Recommended values (used in our environment)” on page 38.

3.12 Troubleshooting

Some of the problems you may encounter when configuring the platform are described in this section, together with our recommendations for solving these issues.

3.12.1 ESS Specialist does not list the WWPN

Each Fibre Channel adapter on the nodes has a unique World Wide Number (Identified in ESS Specialist as World Wide Port Number - WWPN).

When creating a new host with ESS Specialist, select the appropriate WWPN in a scroll list. If you don't see this WWPN, ensure the following:

- The link is correct. You should see the red laser light emitted by the host adapter on the fiber connected to the Storage Area Network - SAN (FC switch).
- The FC adapters on the nodes and the ESS have the proper link type value of *point-to-point* or *arbitrated loop*. If you are unsure about your network topology, leave the default, which is arbitrated loop.
- The command `cfgmgr -vsl fcs0` has been run on the nodes.

For details, refer to 3.9.2, "Configuring logical disks" on page 46.

3.12.2 HACMP does not synchronize

Reboot

After the HACMP code installation, the nodes have to be *rebooted* before configuring and synchronizing the cluster. To find out when the system was last rebooted, use the **`uptime`** command or check the error log (errpt). To reboot: **`shutdown -Fr`** (stop all applications first, and make sure the node is not used by other users).

Refer to "Post-install environment configuration" on page 60.

Check for the same HACMP levels across the cluster

If the HACMP fileset levels differ between the nodes, HACMP may not work properly. Check that the installed filesets are the same on all the nodes, and at the same release level.

To display the HACMP fileset levels, **`lslpp -l |grep cluster*`**

Refer to "Product installation" on page 59 for the complete procedure.

Hosts equivalence

HACMP issues commands on the remote nodes using **`rsh`**. For these remote commands to be accepted on the local node, you have to set up the hosts equivalences.

Refer to 3.7.2, "Enabling remote command execution" on page 36.

3.12.3 HACMP does not start

Check cluster synchronization

After any cluster configuration operation, the cluster must be synchronized.

Logs

You can check what happens with HACMP by viewing the log files, for example:

```
tail -f /tmp/hacmp.out
```

Refer to "Logs" on page 79.

3.12.4 The HACMP clstat command does not work

Clinfo daemon started?

The **clstat** command retrieves the information from the clinfo daemon. This daemon is not started automatically. It is an option in the SMIT HACMP startup menu (**smitty clstart**). If it is not running, stop HACMP on this node, and start it up again with clinfo activated.

To check the status of the HACMP processes: **lssrc -a | grep -E "ES|svcs"**

Refer to "Starting HACMP" on page 75 for more details.

SNMP version in AIX 5L version 5.2

AIX 5L version 5.2 uses SNMP version 3 agents, whereas HACMP uses a SNMP version 1 configuration. Both clinfo and CSPOC require SNMP version 1. With the default SNMP agent, the clinfo daemon is not able to supply cluster information, thus the **clinfo** command will not work properly.

Run the script provided in Example 3-38 to switch to version 1 agents.

Example 3-38 Switch SNMP agents to version 1

```
#!/bin/ksh
LOG=/var/adm/chg_snmpd.log
touch $LOG
/usr/es/sbin/cluster/utilities/clstop -N -g >> $LOG 2>&1
sleep 5
for i in muxatmd aixmibd snmpmibd hostmibd snmpd
do
stopsrc -s $i >> $LOG 2>&1
sleep 2
done
ln -sf /usr/sbin/snmpdv1 /usr/sbin/snmpd
for i in snmpd hostmibd snmpmibd aixmibd muxatmd
do
startsrc -s $i >> $LOG 2>&1
sleep 2
done
sleep 10
/usr/es/sbin/cluster/etc/rc.cluster -N -i >> $LOG 2>&1
sleep 5
```

You can also use the following command to change the running SNMP configuration back to version 1:

```
/usr/sbin/snmpv3_ssw -1
```

For security reasons do not use the default "public" SNMP community. You can change the community name in the /etc/snmpd.conf file, then stop cluster services (incl. clinfo) and specify this new name to clinfo: **chssys -s clinfo(ES) -a "-c NEW_COMMUNITY_NAME"** , then restart HACMP.

3.12.5 Oracle9*i* RAC does not start

HACMP started

Before starting Oracle9*i* RAC instances, the HACMP cluster must be up and running. Check the cluster state with the **/usr/sbin/cluster/clstat** command.

Hagsuser group

These tests must be performed on all nodes:

- Check that the Oracle user is part of the hagsuser group. The name of this group is mandatory, and cannot be changed.
- Check, and change if necessary, the permissions on the **cldomain** executable. This program must be executable by everybody (user, group, other):

  ```
  # chmod a+x /usr/es/sbin/cluster/utilities/cldomain
  ```

- Check, and change the group to *hagsuser* if necessary, for the svcsdsocket.oracle9irac socket file: (assuming that oracle9irac is the name of your cluster, returned by the **cldomain** command):

  ```
  {node2:root}/-> chgrp hagsuser /var/ha/soc/grpsvcsdsocket.oracle9irac
  ```

- Check, and change if necessary, the group permissions for the grpsvcsdsocket.oracle9irac socket:

  ```
  {node2:root}/-> chmod g+w /var/ha/soc/grpsvcsdsocket.oracle9irac
  ```

The HAGS socket file needs to be writable by the Oracle user and the **cldomain** executable needs to be executable by Oracle. By configuring the group and permissions for the grpsvcsdsocket.oracle9irac file, the instance will be able to communicate with HAGS and the instance will mount.

See Example 3-39 for the valid permissions and group membership for the socket files used by Oracle9*i* RAC.

Example 3-39 Permissions on the /var/ha/soc directory

```
{node2:root}/var/ha/soc-> ls -l
total 0
srw-rw-rw-   1 root     haemrm            0 Jun 10 18:18 em.clsrv.oracle9irac
srw-rw----   1 root     haemrm            0 Jun 10 18:18 em.rmsrv.oracle9irac
drwxrwxrwx   2 root     system          256 Jun 11 19:20 grpsvcs.clients.oracle9irac
srw-rw-rw-   1 root     hagsuser          0 Jun 10 18:18 grpsvcsdsocket.oracle9irac
drwxrwx---   2 root     haemrm          256 Jun 10 18:18 haem
drwxr-xr-x   2 root     system          256 May 13 17:54 hats
drwxr-xr-x   2 root     system          256 Jun 10 18:18 topsvcs
```

UDP tuning

The udp_sendspace value must always be greater than Oracle9*i* RAC's db_block_size. Otherwise Oracle9*i* RAC will not start.

Refer to 3.7.3, "Tuning network options" on page 38 for more information about the network parameters.

3.12.6 GPFS issues

GPFS requires a quorum of nodes to run. It also requires that both nodes holding the GPFS configuration data be up and running for any configuration changes to be performed.

In case of a system restart, GPFS will be started automatically and the GPFS file systems will be mounted.

To shut down and unmount the GPFS file systems on all nodes, issue:

```
# mmshutdown -a
```

To start GPFS and mount the GPFS file systems on all nodes, issue:

```
# mmstartup -a
```

3.12.7 Miscellaneous

Check the AIX error report:

```
{node2:root}/-> errpt
```

4

Oracle9*i* RAC installation and configuration

This chapter describes the Oracle9*i* RAC installation and configuration procedures in an IBM eServer pSeries server cluster environment. We describe the pre-installation tasks, the installation steps for Oracle9*i* RAC (using Oracle Universal Installer (OUI)), and the post-installation tasks we recommend to check and validate installation.

The Oracle9*i* RAC version for UNIX is 9.2.0.1, with 9.2.0.3 patch level, running on IBM AIX pSeries (RS/6000). Oracle9*i* RAC runs on AIX versions 5.1 or 5.2, with either a 64-bit or 32-bit kernel. Oracle9*i* RAC is 64-bit code, thus requiring 64-bit hardware.

4.1 Prerequisites and dependencies

To ensure a successful installation, there are several aspects you should consider in the planning phase, such as HW and SW interoperability and dependencies.

4.1.1 OS prerequisites checking

The pre-installation steps described in this section must be performed on all the nodes associated with an Oracle9*i* RAC cluster. The minimal software required for GPFS/RAC installation is AIX 5.1 ML01. In addition we need PTF IY28111. For AIX 5.2, ML01 (also known as 5.2B) is required.

The following operating system prerequisites are described in 3.6, “Node installation and configuration” on page 30:

- Operating system level
- HACMP filesets
- GPFS filesets
- Java filesets

In addition to the software prerequisites, the following files have to be checked:

- Verify in the /etc/hosts.equiv file on all cluster nodes for user *oracle* entries.
- Check the ~/.rhost file for user *oracle*. During installation and for normal operations, Oracle processes require remote common access to all nodes in the cluster.
- Check if the following tools exist, and if they are executable:
  - make
  - ld
  - ar
  - nm

Before starting any Oracle installation, the HACMP cluster must be defined, synchronized, and running on all nodes. Private networks have to be defined for Oracle interconnect traffic. See Example 4-1 for cluster status and configuration. Also check if the GPFS daemons are active and the GPFS file systems are mounted.

Example 4-1 Checking HACMP and GPFS cluster configuration

```
{node1:root}/-> lssrc -a|egrep "ES|svcs"
 topsvcs          topsvcs          569398       active
 grpsvcs          grpsvcs          565294       active
 emsvcs           emsvcs           581672       active
 emaixos          emsvcs           426070       active
 clstrmgrES       cluster          598070       active
 clsmuxpdES       cluster          475148       active
 clinfoES         cluster          557092       active
 grpglsm          grpsvcs                       inoperative

{node1:root}/-> /usr/es/sbin/cluster/utilities/cllsif |grep private
onet11   service    ora_interconnect1 ether   private    node1      10.10.10.61  en1
onet21   service    ora_interconnect2 ether   private    node1      172.16.10.61 en2
onet12   service    ora_interconnect1 ether   private    node2      10.10.10.62  en1
onet22   service    ora_interconnect2 ether   private    node2      172.16.10.62 en2
onet13   service    ora_interconnect1 ether   private    node3      10.10.10.63  en1
onet23   service    ora_interconnect2 ether   private    node3      172.16.10.63 en2
onet14   service    ora_interconnect1 ether   private    node4      10.10.10.64  en1
```

```
onet24   service    ora_interconnect2 ether   private    node4      172.16.10.64 en2

{node1:root}/-> lssrc -a|grep mmfs
   mmfs             aixmm            581720       active
{node1:root}/-> df
Filesystem    512-blocks      Free %Used    Iused %Iused Mounted on
------------------<<Ommited lines>>------------------------
   /dev/data       8621440      44736     4%     4096 2% /data
   /dev/oracle     8621440      44736     4%     4096 2% /oracle
{node1:root}/->
```

Defining an Oracle administrative user and group

- As user root, create an installation and administrative group for Oracle (we created the group “dba”). Repeat this on all nodes in the cluster. Make sure the group has the same ID on all nodes.
- Create the hagsuser group (mandatory name). This group is needed for Oracle to communicate with the cluster subsystem via the socket file located in /var/ha/soc/grpsvcsd.oracle9irac. This socket file must have group permission set to *hagsuser* in order to allow Oracle to communicate with cluster group services (if this is not configured, the rootpre.sh script will fail). Repeat on all nodes in the cluster. Make sure the group has the same ID on all nodes.
- Create the /oracle/home directory, then create user *oracle*, with primary group dba and secondary group hagsuser. Repeat on all nodes in the cluster. Make sure the user has the same ID on all nodes.

  We decided to store the home directory for user oracle on a GPFS file system, so we did not need to maintain multiple home directories for this user.

  Set the password for user oracle to your choice, then propagate the change to all nodes.
- Set the umask for user oracle to 022.

Example 4-2 Oracle administrative user and group creation

```
{node1:root}/-> mkdir /oracle/home
{node1:root}/-> mkgroup -'A' id='9203' dba
{node1:root}/-> mkgroup -'A' id='9000' hagsuser
{node1:root}/-> mkuser  id='9203' pgrp='dba' groups='hagsuser' home='/oracle/home' oracle
```

Note: The oracle user ID must be less than 65536 for Real Application Cluster.

Oracle user limits

Modify the operating system limits for user oracle. Depending on your database application, this user may require “unlimited” for all parameters, except stack, coredump and nofiles.

Check the /etc/security/limits file and verify the settings, as shown in Example 4-3.

Example 4-3 oracle user limits

```
{node4:oracle}/oracle/home-> ulimit -a
time(seconds)        unlimited
file(blocks)         unlimited
data(kbytes)         unlimited
stack(kbytes)        2097152
memory(kbytes)       unlimited
coredump(blocks)     2097151
```

```
nofiles(descriptors) 2000
```

Create storage space for Oracle9i RAC installation code

Log in as user root and create a file system large enough to store the content of the four Oracle9*i* RAC installation CDs. Follow the instructions shown in Example 4-4.

Example 4-4 Oracle9i RAC installation repository

```
{node1:root}/-> mklv -t jfs2 -y oraclecd_lv rootvg 150 (assuming LP size=32MB)
{node1:root}/-> crfs -v jfs2 -p rw -d /dev/oraclecd_lv -m /oraclecd -A yes
{node1:root}/-> mount /oraclecd
{node1:root}/-> mkdir /oraclecd/Disk1 /oraclecd/Disk2 /oraclecd/Disk3 /oraclecd/Disk4

Copy all the information from each Oracle install CD into the corresponding directory, than
change ownership to oracle user:

{node1:root}/-> chown -R oracle.dba /oraclecd
```

Performing this step will allow Oracle9*i* RAC installation without changing CDs during the installation process.

Check system parameters

For Oracle9*i* RAC 9.2.0.x it is not necessary to change the default kernel parameters prior to installation. However, some of the system parameters may be increased according to your storage configuration, as shown in Example 4-5.

Example 4-5 Kernel parameters

```
{node1:root}/-> lsattr -El aio0
autoconfig available STATE to be configured at system restart True
fastpath   enable    State of fast path                       True
kprocprio  39        Server PRIORITY                          True
maxreqs    4096      Maximum number of REQUESTS               True
maxservers 60        MAXIMUM number of servers per cpu        True
minservers 30        MINIMUM number of servers                True
```

Check network options

Check network option parameters according to 3.7.3, “Tuning network options” on page 38. These parameters have to be increased due to communication needs of the Oracle interconnect network.

Always check the latest Oracle9*i* RAC documentation and release notes:

- *Oracle9i Installation Guide Release 2 (9.2.0.1.0) for UNIX Systems: AIX-Based Systems, Compaq Tru64 UNIX, HP 9000 Series HP-UX, Linux Intel, and Sun Solaris.*
- *Oracle9i Release Notes -Release 2 (9.2.0.1.0) for AIX-Based 5L Systems (64-bit).*

The PDF version of these manuals can be downloaded from:

```
http://otn.oracle.com/docs/products/oracle9i/content.html
```

Note: You must register to access this site.

4.2 Oracle9*i* RAC installation and configuration (on GPFS)

In a GPFS file system environment you have two choices for installing Oracle code:

1. Install the code on each node (non-shared file system)

   Create a JFS or JFS2 file system on each node and set oracle.dna ownership on these file systems. When installing Real Application Clusters, the Oracle Universal Installer (OUI) will copy the Oracle code from the node from which you are running the installer to the other nodes in the cluster. This results in one copy of the Oracle binaries on each node.

2. Install the Oracle software on a GPFS file system, thus creating only one copy of the Oracle binaries. All instances will share the same code. This option is more convenient because all configuration and SW maintenance can be performed from any node in the cluster.

Single point of control instance management

On each node, create the directory /var/opt/oracle and set the ownership to the user oracle. During Oracle9*i* RAC installation, a file called srvConfig.loc will be created in this directory.

This file is used for specifying the destination of the Oracle server manager utility (srvctl) common configuration file. This file must be on a shared file system (GPFS), and is used by the Oracle server manager for central management of the Oracle9*i* RAC instances.

Oracle code file system configuration

In our environment we chose a GPFS file system to store the Oracle code. For GPFS file system configuration, refer to 3.10.2, “GPFS cluster configuration” on page 52.

Check if the designated Oracle code file system is mounted. We used /oracle as mount point for this file system. All binaries, Oracle logs, and initialization (init) files will be on the GPFS file system. Be sure to allocate enough space on this file system (see Example 4-6), considering future SW upgrades and Oracle log files.

Example 4-6 File system allocation

```
{node1:root}/-> df -Pk
Filesystem    1024-blocks      Used Available Capacity Mounted on
/dev/hd4            65536     37936     27600      58% /
/dev/hd2          1376256   1354380     21876      99% /usr
/dev/hd9var        131072     66216     64856      51% /var
/dev/hd3           917504     49744    867760       6% /tmp
/dev/hd1            65536     15132     50404      24% /home
/proc                   -         -         -       -  /proc
/dev/hd10opt       131072     57436     73636      44% /opt
/dev/data       144424960 129734240  14690720      90% /data
/dev/oracle       9748480   6607360   3141120      68% /oracle
```

Change the ownership and access permission for the Oracle directory:

```
chown -R oracle.dba /oracle
```

Oracle user environment

Set up the Oracle environment in the $HOME/.profile file of the user oracle. Depending on your configuration, you may choose the user oracle home directory on an internal disk (/home/oracle, in which case you have to propagate the same environment on all nodes), or on a GPFS file system, in the /oracle directory (/oracle/home in our environment).

Example 4-7 Setting environment variables in ~/.profile

```
>>> Previous lines are generic environment lines (MAILMESG, PS1 etc.,)<<<<
# Oracle specific environment starts HERE
HOST=`hostname -s`
# This stanza selects the value of ORACLE_SID variable depending on the host the oracle
```

```
# user logs in.
case ${HOST} in
        node1)
        SID=1;;
        node2)
        SID=2;;
        node3)
        SID=3;;
        node4)
        SID=4;;
esac
# Variables needed during installation and normal operation
export ORACLE_SID=rac${SID}
export DISPLAY=node1:0.0
export TMPDIR=/oracle/temp
export TEMP=/oracle/temp
export ORACLE_BASE=/oracle
export ORACLE_HOME=/oracle/product/9.2.0
export PATH=$ORACLE_HOME/bin:$PATH
```

The following variables are mandatory to perform an Oracle9*i* RAC installation:

- PATH.
- ORACLE_SID is the system identifier for an Oracle server instance. This variable uniquely identifies a database instance. For consistency, we chose RAC as instance name prefix (see Example 4-7 on page 89 and Table 4-1).

Table 4-1 Instance name selection

Host name	Node name	Thread ID	SID
node1	node1	1	rac1
node2	node2	2	rac2
node3	node3	3	rac3
node4	node4	4	rac4

- ORACLE_HOME is the directory that contains the Oracle software (binaries, libraries etc.).
- TMPDIR and TMP: during installation Oracle needs approx. 800 MB of temporary space. For maintaining the system /tmp directory under control, we allocated a separate temporary space in the /oracle/temp directory, and assigned the two variables to point to this directory.
- ORACLE_BASE specifies the base directory for Oracle software.

Also, the DISPLAY variable is needed for nodes that do not have a graphical display.

Verify the oracle user environment by logging in to the systems (as user oracle) and displaying the variables, as shown in Example 4-8.

Example 4-8 Testing the environment

```
{node1:oracle}/oracle/home-> echo $ORACLE_SID
rac1
```

Now start the Oracle9i Universal Installer (OUI) graphical interface tool.

4.2.1 Running Universal Installer for Oracle9*i*

Log in as user oracle on node1 and start the Oracle Universal Installer, as shown in Example 4-9.

Example 4-9 Starting Oracle Universal Installer (OUI)

```
{node1:oracle}/-> export DISPLAY=node1:0
{node1:oracle}/-> cd /oraclecd/Disk1
{node1:oracle}/oraclecd/Disk1-> su root
{node1:oracle}/oraclecd/Disk1-> ./rootpre.sh
>>>>> Answer the questions .......<<<<<<<
{node1:oracle}/oraclecd/Disk1-> exit
{node1:oracle}/oraclecd/Disk1->
{node1:oracle}/oraclecd/Disk1-> ./runInstaller
```

We recommend that you run the rootpre.sh script before starting the graphical interface. This script checks for the cluster environment (to enable Real Application Cluster installation).

If this script returns an error (exit code not 0), check the groups and permissions, make the necessary corrections, and run it again.

Oracle9*i* RAC version 9.2.0.x on AIX 5L, running a 64-bit kernel, does not load the post-wait kernel extension.

The Welcome window is displayed; see Figure 4-1.

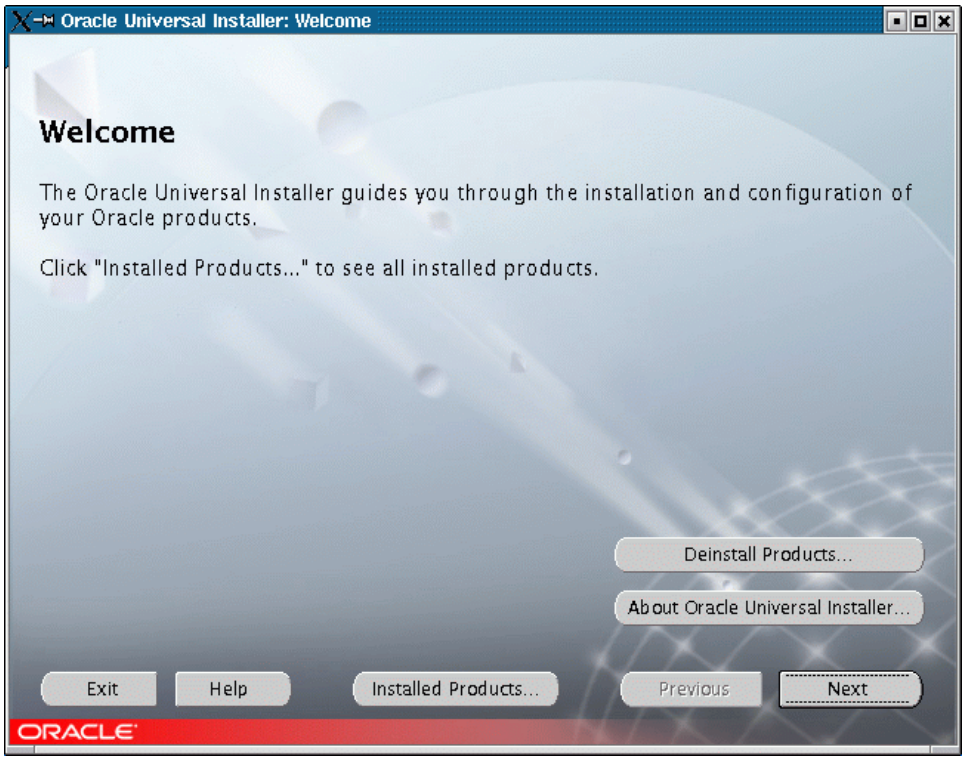


Figure 4-1 The Oracle9i Welcome window

Click **Next** to display Inventory Location (as in Figure 4-2 on page 92).

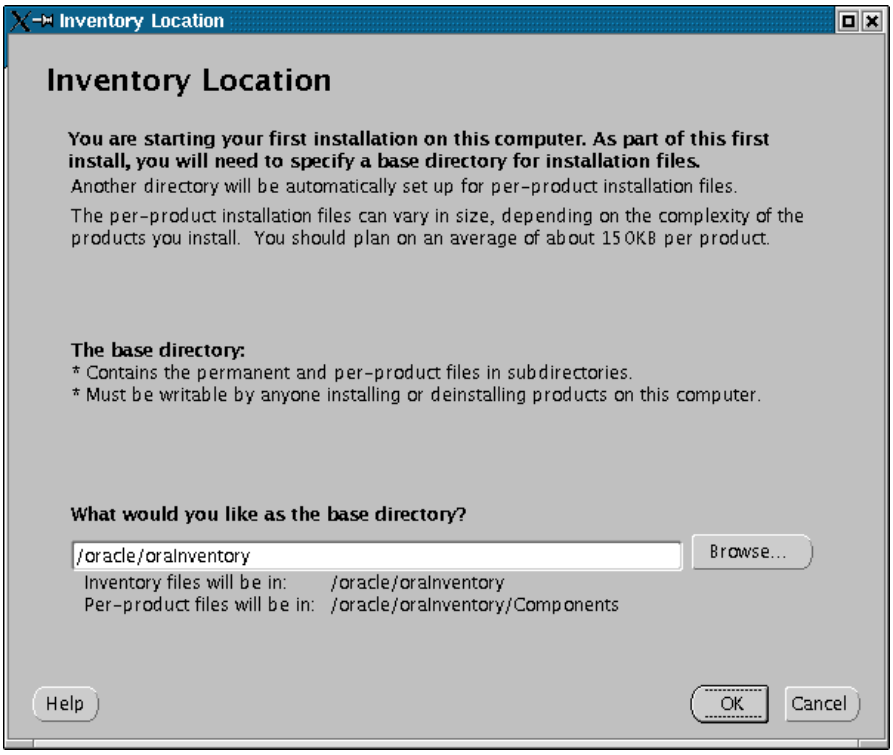


Figure 4-2 Inventory location window

In the next screen (Figure 4-3), enter the group for user oracle. The dba group provides the authorization for starting and stopping the database (and other administrative tasks).

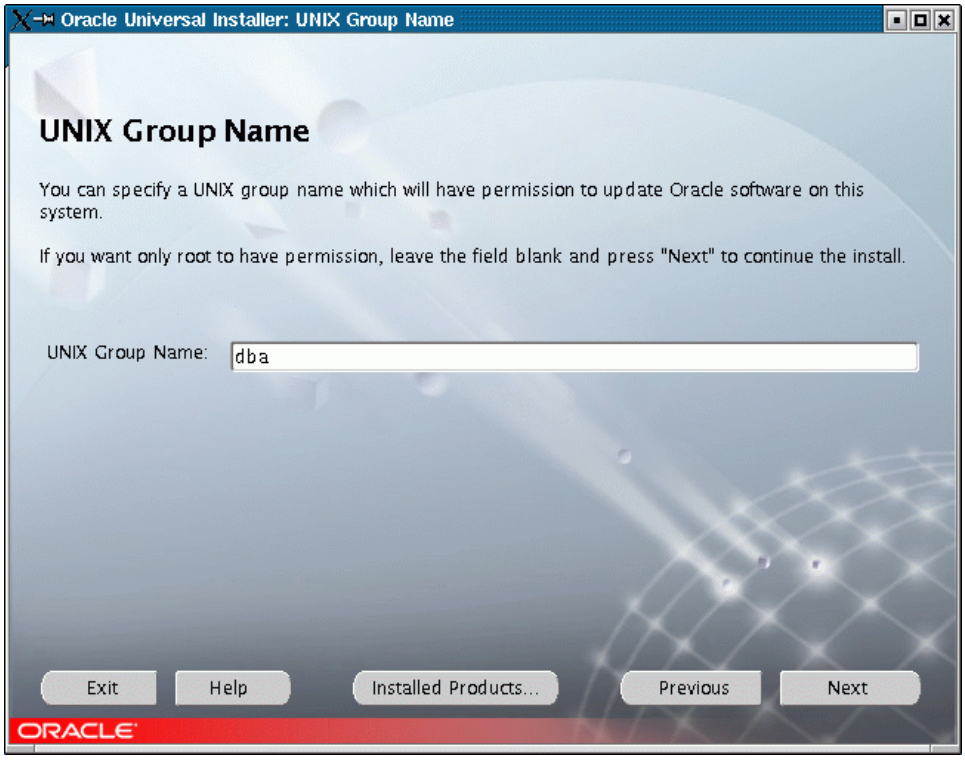


Figure 4-3 UNIX group name window

A pop-up window (shown in Figure 4-4) asks to execute /tmp/orainsRoot.sh as root user.

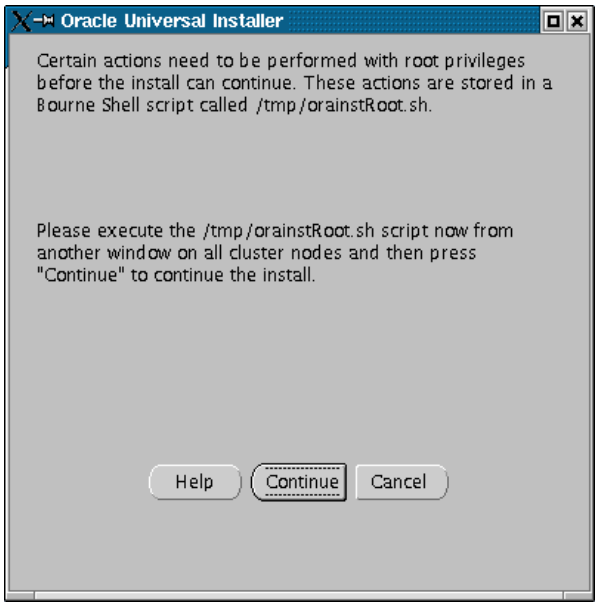


Figure 4-4 The pop-up window - Execute orainstRoot.sh

Open another terminal window, log in as root user, and execute /tmp/oraintRoot.sh. This creates the /etc/oraInst.loc file, and changes the group name for the /oracle/oraInventory directory to dba.

Continue on the window shown in Figure 4-5 to display the Cluster Node Selection.

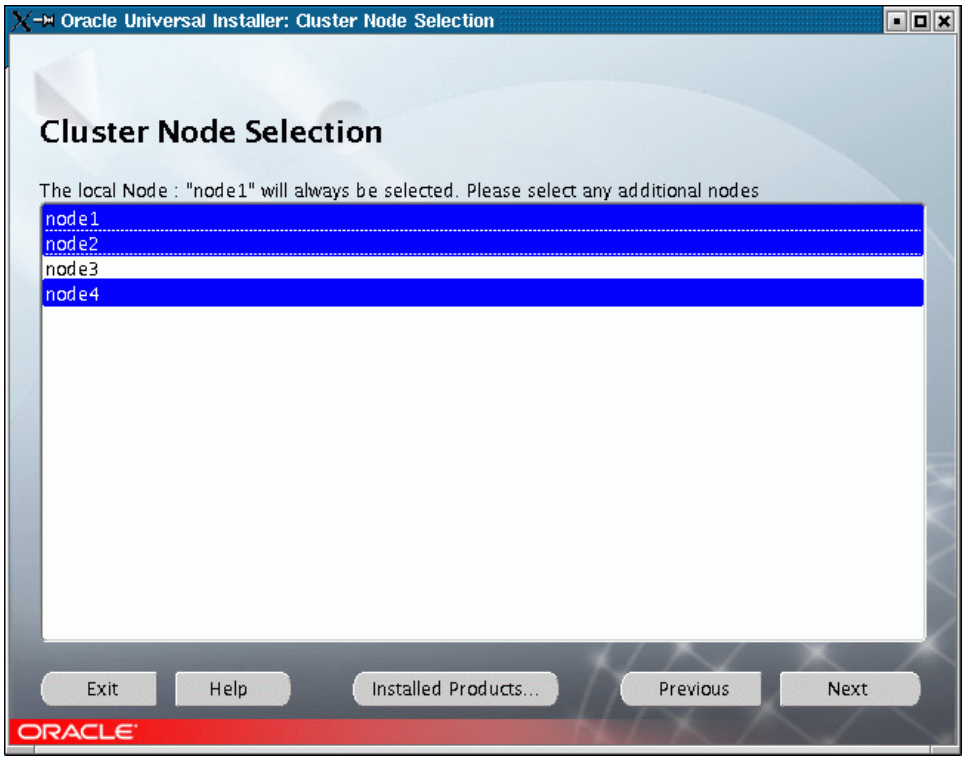


Figure 4-5 The node selection window

Select the nodes that are to be part of the cluster.

Note: Cluster membership is provided by HACMP. Thus, only the nodes defined inside the same HACMP cluster will be available for RAC installation.

If the installer does not show the Node Selection window, do not go further with the installation. Stop, diagnose the problem by executing the **`lsnodes -v`** command, and check your cluster configuration. Then correct the problem and restart installation.

This command is located in the $TEMPDIR/OraInstall directory. The local node is always selected. Click **Next** to display the File Locations window

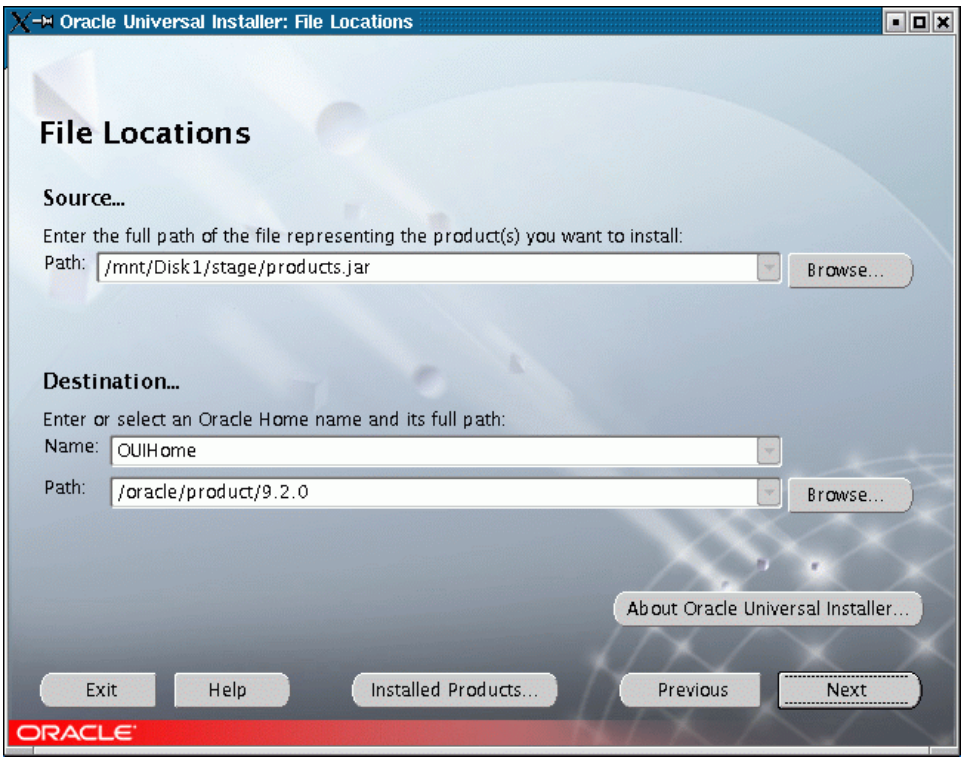


Figure 4-6 File Locations window

Change or verify the products you want to install and click **Next** to display the Available Products window and select **Oracle9*i* Database** (see Figure 4-7 on page 95).

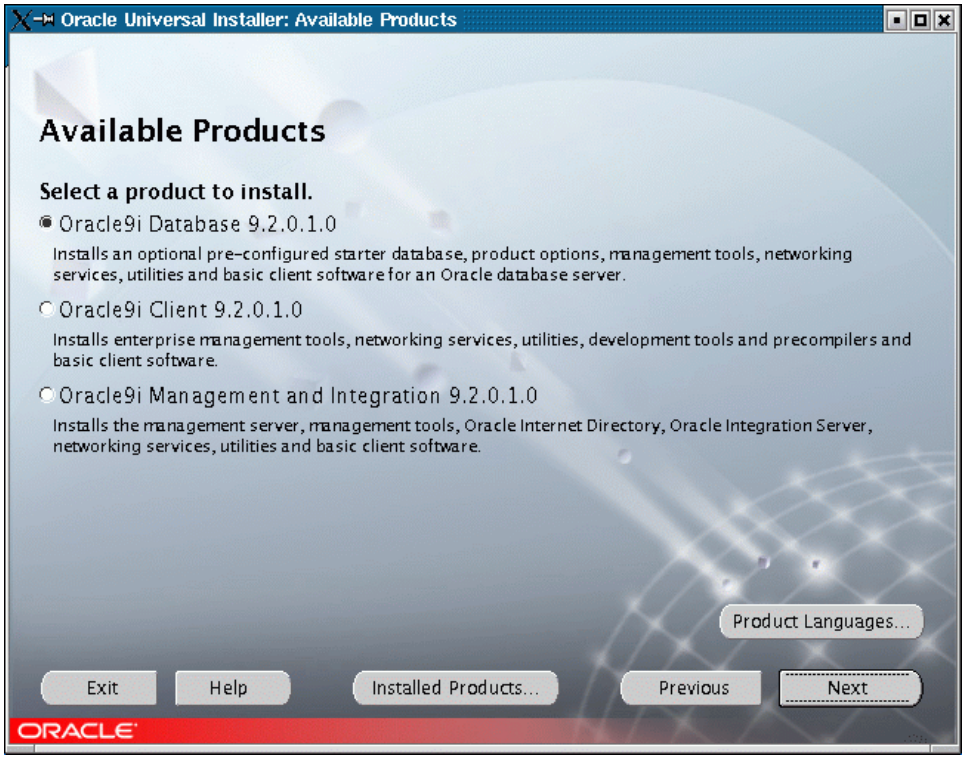


Figure 4-7 Available Products window

In the next menu, select the installation type. We selected **Custom** installation (see Figure 4-8).

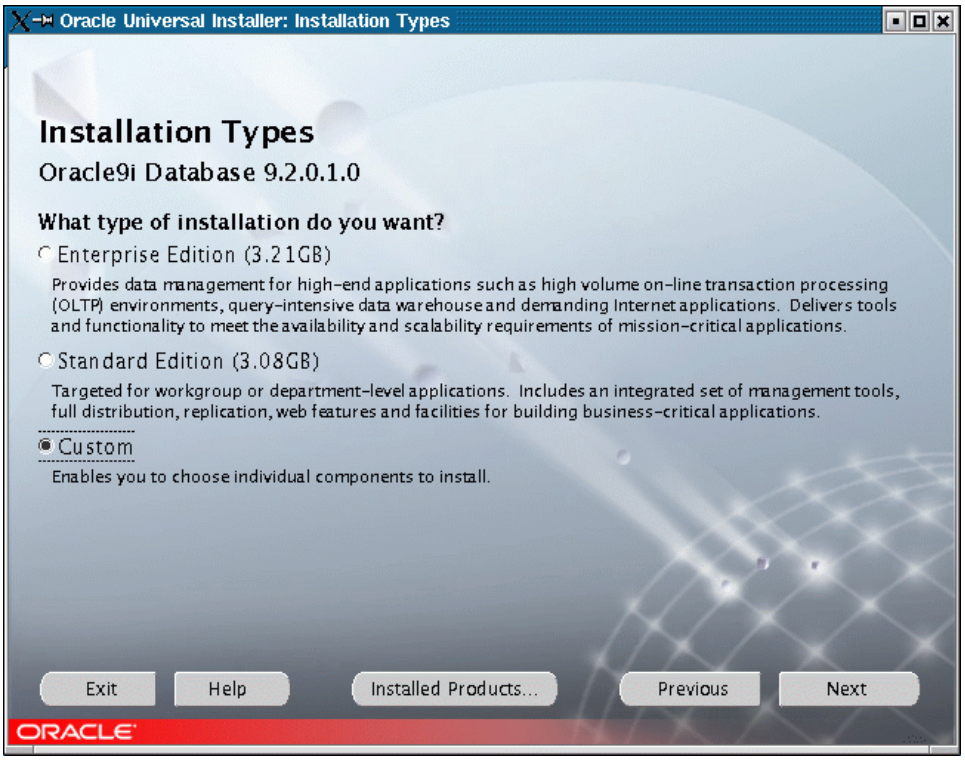


Figure 4-8 Product Installation Types window

In the next window, select the Available Product Components (see Figure 4-9).

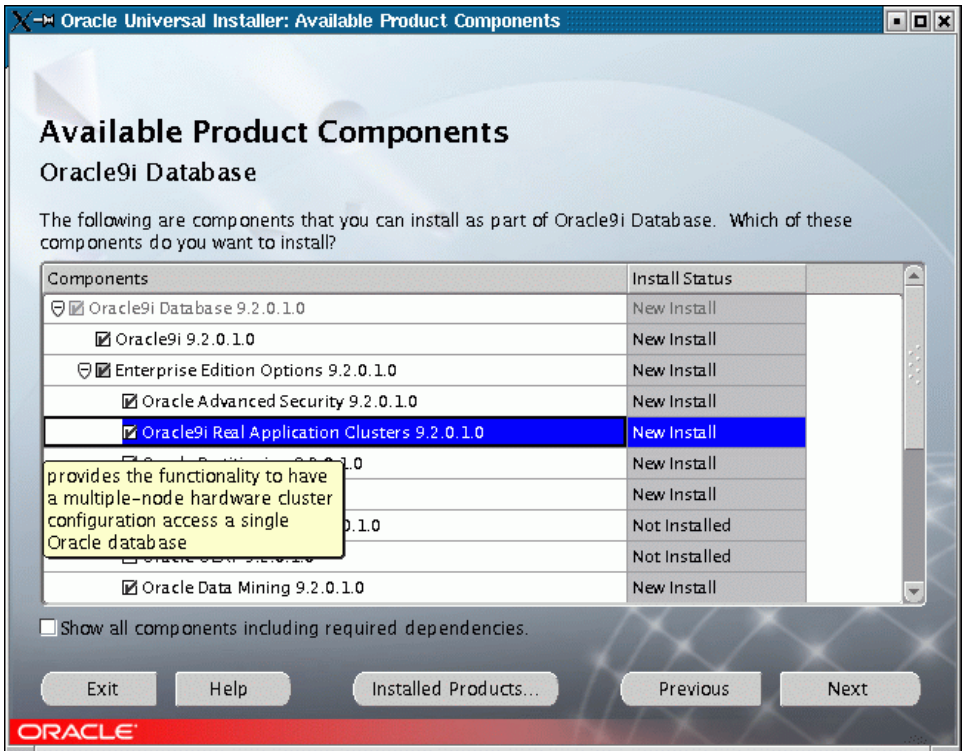


Figure 4-9 Available Product Components

Select or deselect items in the list according to your installation requirements. Make sure the following mandatory products are selected:

- Oracle9i 9.2.0.1
- Oracle9i Application Cluster 9.2.0.1

Proceed to the next display, shown in Figure 4-10 on page 97. Correct the installation destination directory, if necessary.

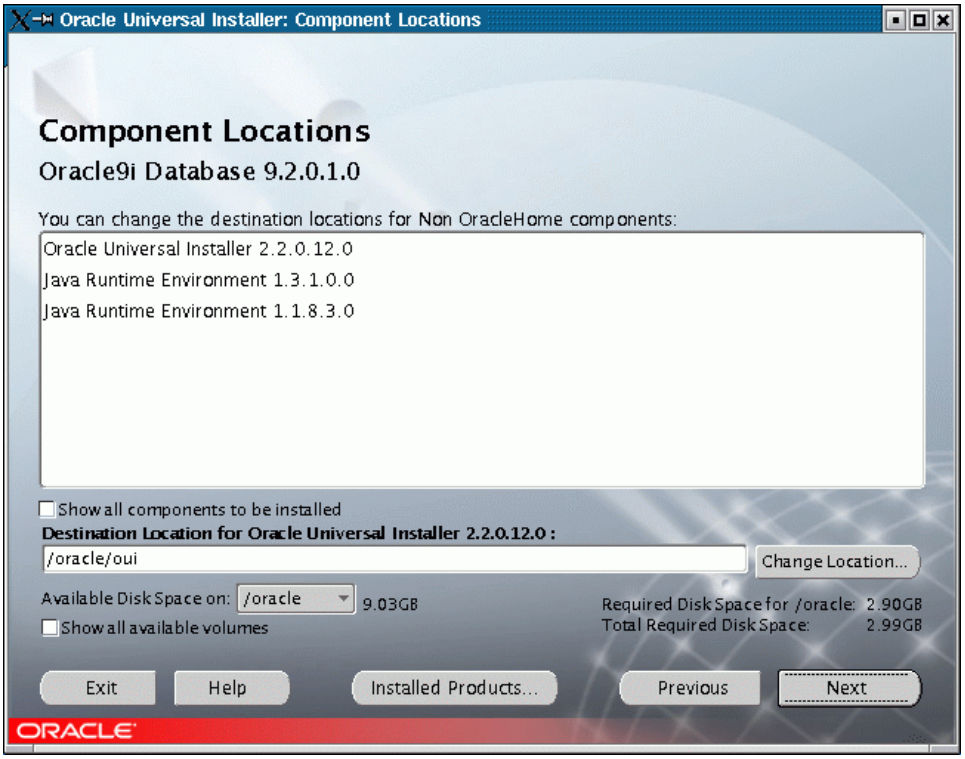


Figure 4-10 Component Locations window

Next enter the server manager configuration file name, as shown in Figure 4-11 on page 98. We entered `/oracle/itso_rac`. For details, refer to “Single point of control instance management” on page 89.

Note: Be sure the shared configuration file is available in the shared storage space, and can be accessed from all nodes in the cluster.

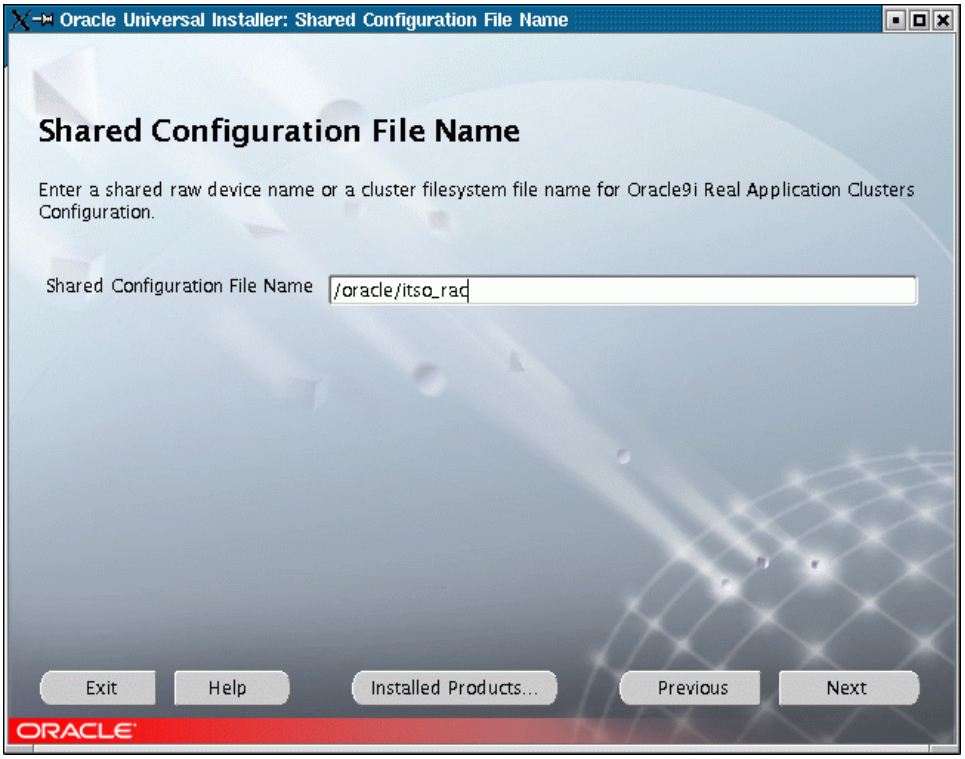


Figure 4-11 Shared Configuration File Name window

Click **Next** to display the Privileged Operation Groups window, and confirm the UNIX group for Database Administrator and Database Operator, as shown in Figure 4-12.

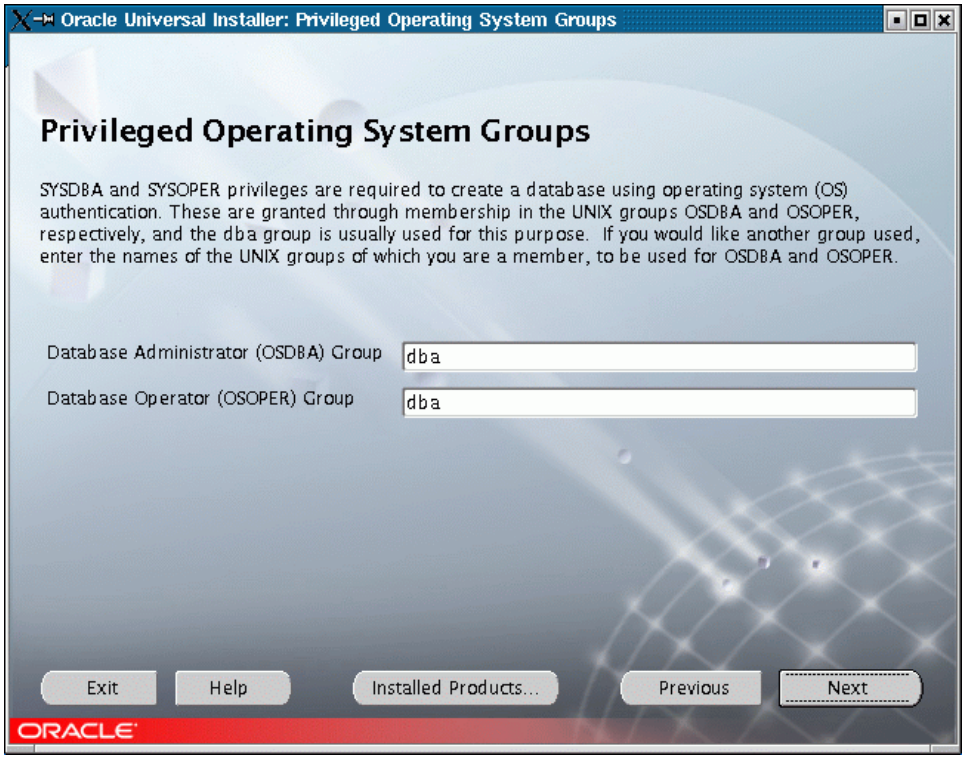


Figure 4-12 Privileged Operating System Groups

Next, choose an existing Oracle Management Server Repository or define a new one. We selected to create a new repository; see Figure 4-13.

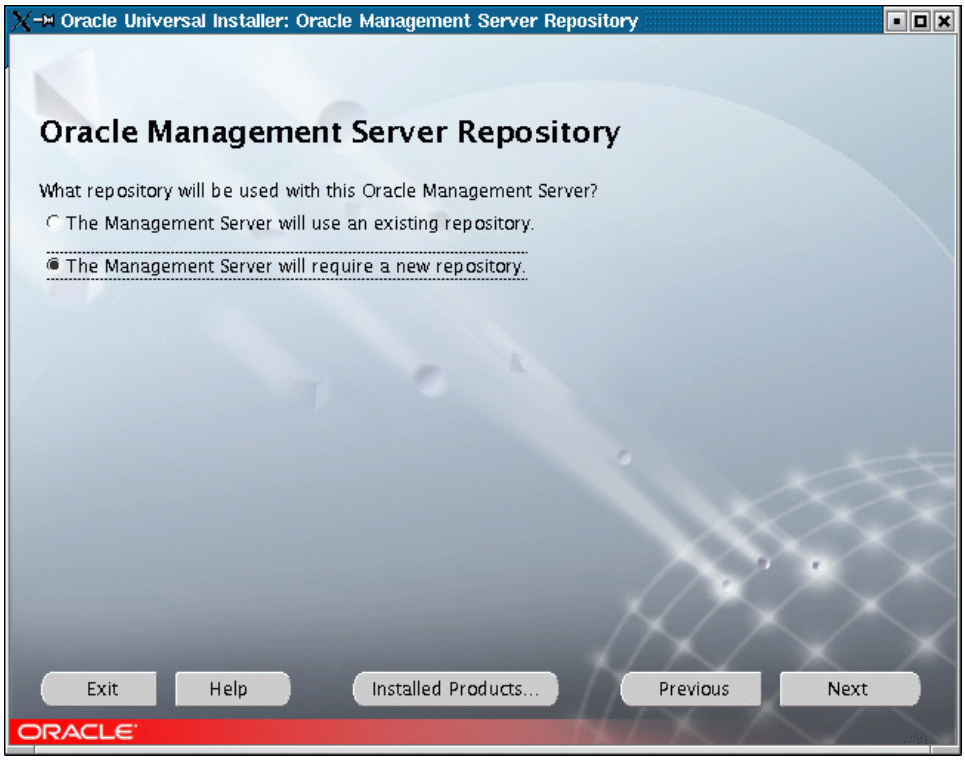


Figure 4-13 Oracle Management Server Repository

Next, the Create Database window is displayed. We chose **No** because we decided to create the database later, after applying the Oracle9*i* RAC 9.2.0.3. patch set.

Next, select the Java Development Kit (JDK) location in the window shown in Figure 4-14 on page 100.

Figure 4-14 Choose JDK Home Directory window

Figure 4-15 shows the overall selections chosen for our installation.

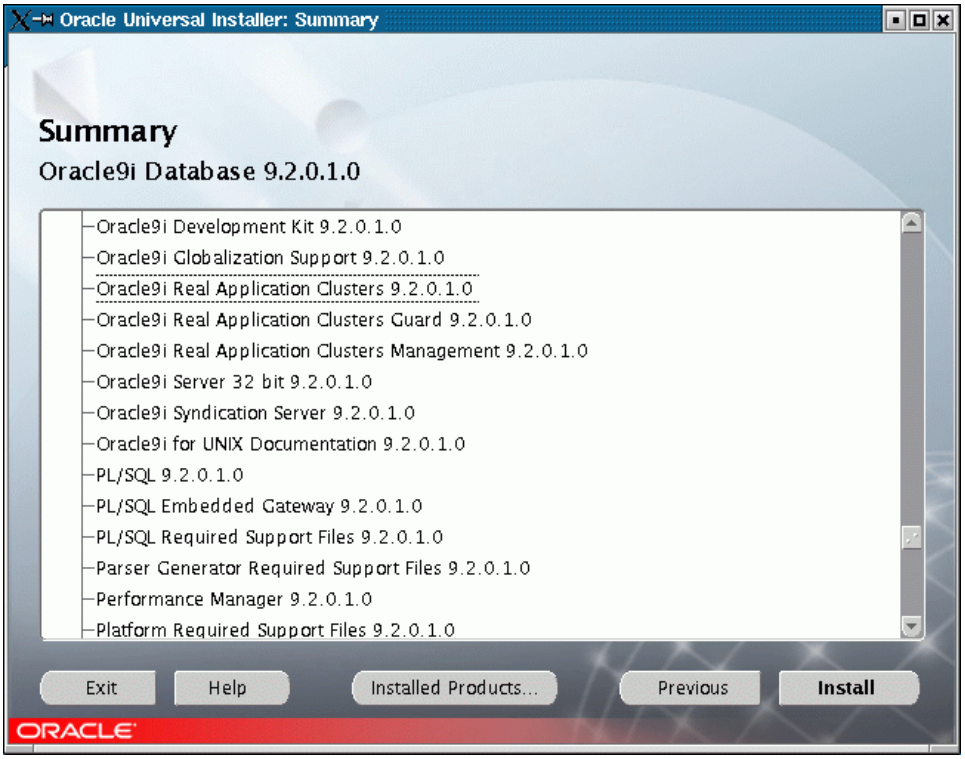


Figure 4-15 Summary window

To verify the Oracle9*i* RAC Real Application Cluster, scroll down the list. Click **Nex**t to start the installation (see Figure 4-16).

Figure 4-16 The Install window

Note: The Oracle9*i* RAC Universal Installer creates a log file of the current install session in the $ORACLE_BASE/oraInventory directory. Check log messages using:

```
tail -f /oracle/oraInventory/logs/installactions.log
```

When installation finishes, a pop-up window appears (Figure 4-17) telling you that you have to execute the $ORACLE_HOME/root.sh script.

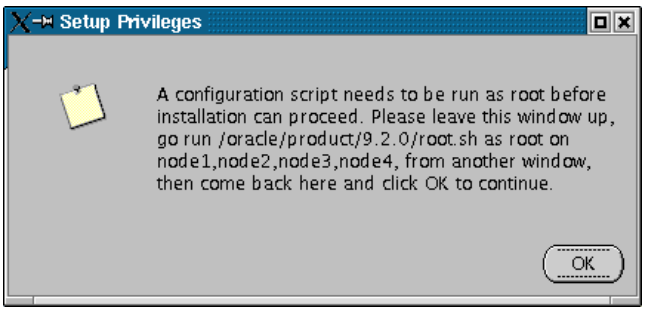


Figure 4-17 Setup privileges pop-up window

In a terminal window, log in as root user with the user oracle environment (su without the "-" command option) and execute $ORACLE_HOME/root.sh.

Click **Installed Products** to check the product installation again, and click **Next** to exit the installation. See Figure 4-18 on page 102.

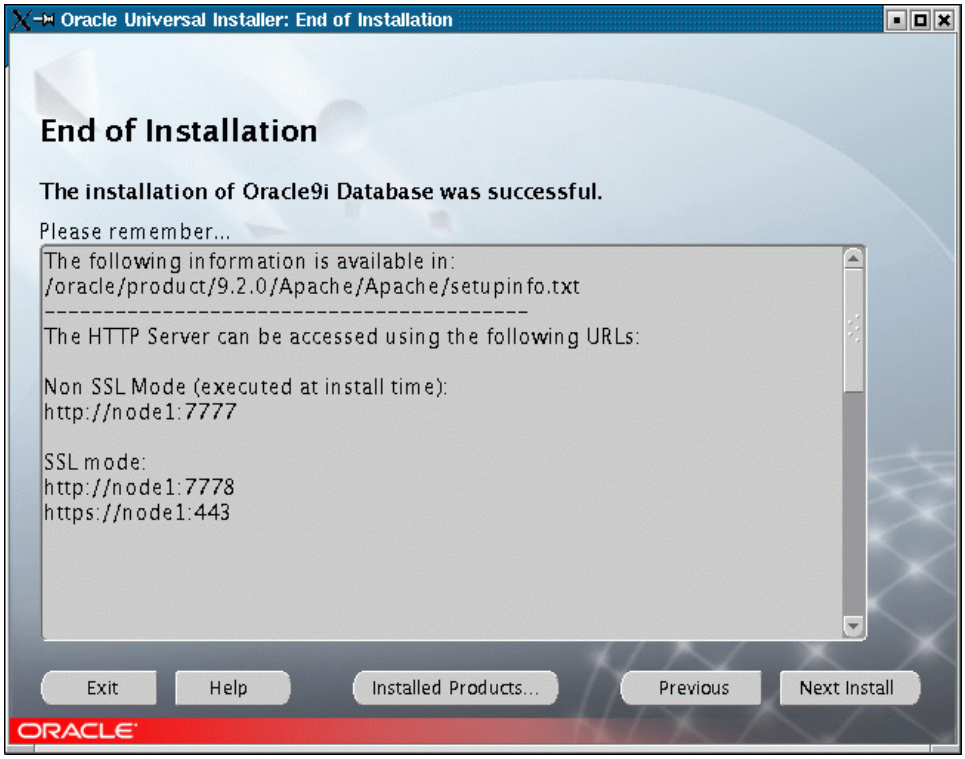


Figure 4-18 End of Installation window

Note: If you chose to have Oracle code on all nodes on non-shared file systems, this step may take a long time. This is because after installation on the local node, OUI copies the code to the other nodes on the cluster using the **rcp** command. Check by monitoring the network activity.

To verify that the installation is ok, test using **sqlplus** (**sqlplus '/ as sysdba'**). If the "SQL>" prompt is displayed, the installation was successful.

4.2.2 Oracle9*i* RAC Database Server Patch set 9.2.0.3.0

Download this patch from the Metalink site or contact Oracle Support to obtain the code on CDROM.

Copy the patch files into the /oraclecd directory. Check the README file to understand all the steps you need to perform.

Start Oracle Universal Installer and choose the nodes on which to apply the patch. The File Locations dialog appears, as shown in Figure 4-19 on page 103.

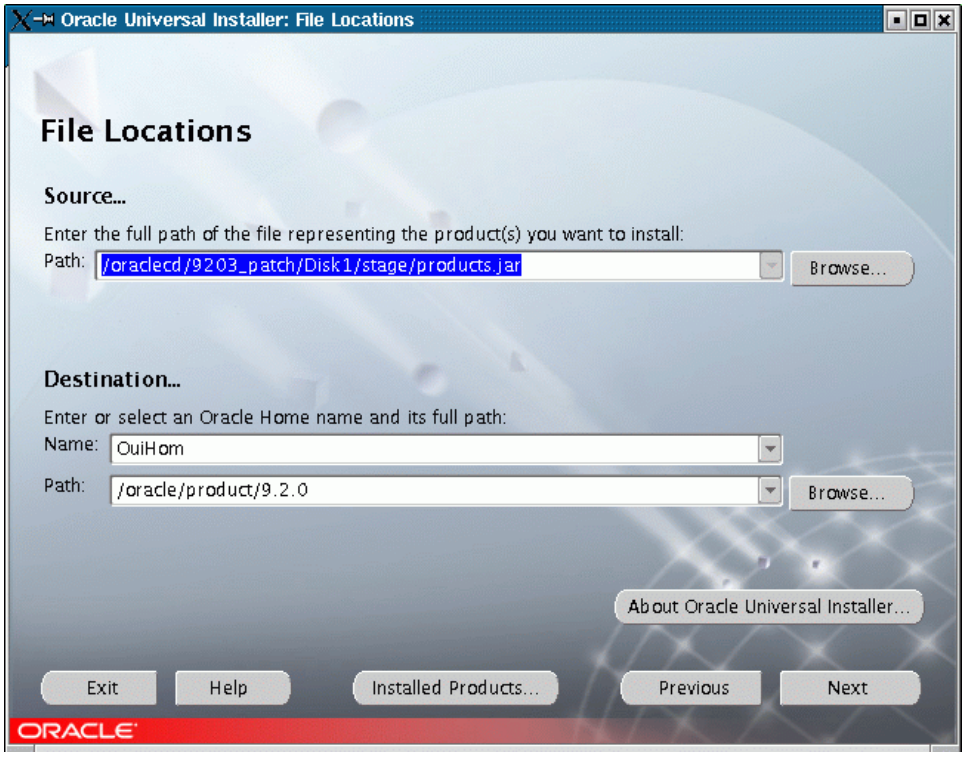


Figure 4-19 Patch File Locations

Click **Next** to show the available updates in the Summary window (see Figure 4-20).

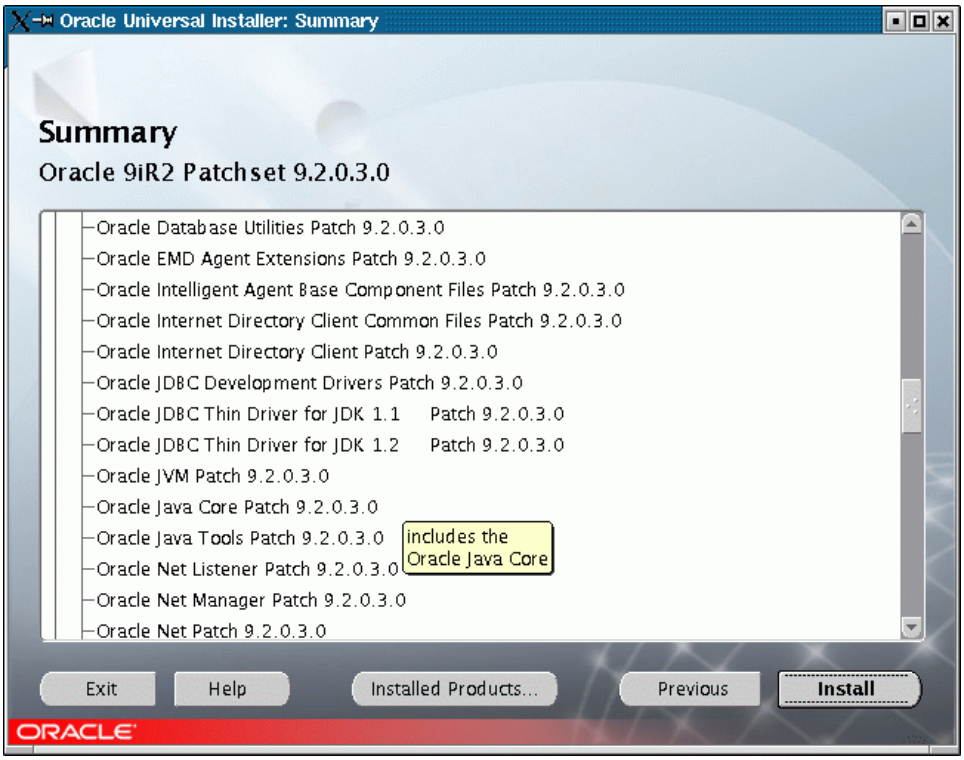


Figure 4-20 Summary Oracle Patch set

4.2.3 Oracle Net Services initial configuration

In this section we describe the Oracle Net Services configuration. The tool used for this purpose is Network Configuration Assistant (**netca**), which creates the listener.ora and tnsnames.ora files, containing basic network components to allow client connections to the database instances.

In this configuration, since we use a shared Oracle home, all nodes will be sharing the same tns and listener files.

When creating a database with **dbca**, the tool discovers the information automatically and updates the network configuration files with the new database information.

Note: To be able to run **netca** after applying patch set 9.2.0.3, make the following updates to the $ORACLE_HOME/bin/netca script:

The line
CLASSPATH=**$NETCAJAR:$NETCFGJAR**:$EWTJAR:$HELPJAR:$SHAREJAR:$JREJAR:$EWTOTHER:$SWINGJAR:$NETTOOLSDIR:$OPSJAR

should be changed to:

CLASSPATH=**$NETCFGJAR:$NETCAJAR**:$EWTJAR:$HELPJAR:$SHAREJAR:$JREJAR:$EWTOTHER:$SWINGJAR:$NETTOOLSDIR:$OPSJAR

A new feature in the Oracle9*i* RAC “service registration” allows you to skip the listener.ora file creation (used for static listener configuration).

Static listener configuration is only required if you plan to use the Oracle Enterprise Manager tool to discover how the listeners are mapped to the database instances.

listener.ora configuration

To configure the listener.ora file, use **netca**.

Login as user oracle using a terminal window, and execute the **netca** command (be sure the proper DISPLAY variable is set); see Figure 4-21.

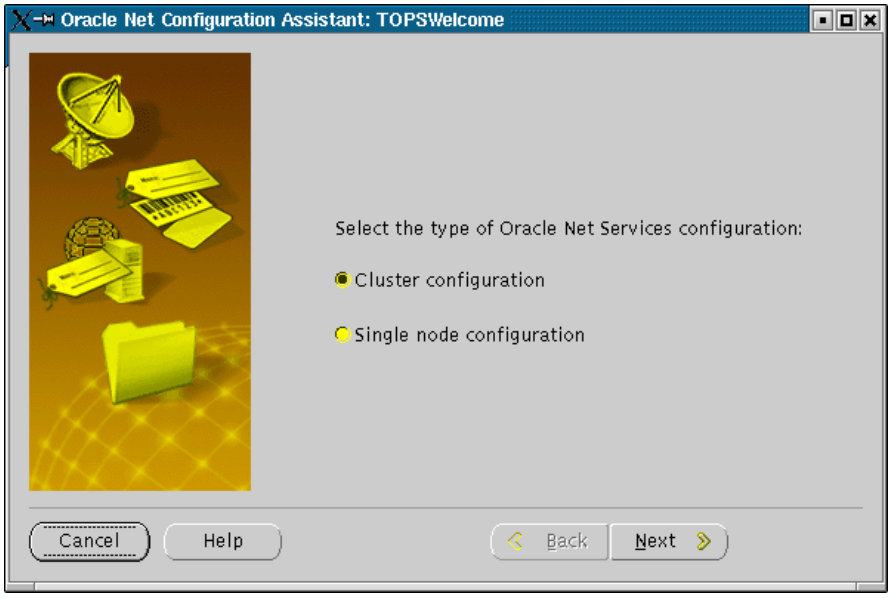


Figure 4-21 Welcome window

Select **Cluster configuration** and click **Next**. Select the nodes, as shown in Figure 4-22.

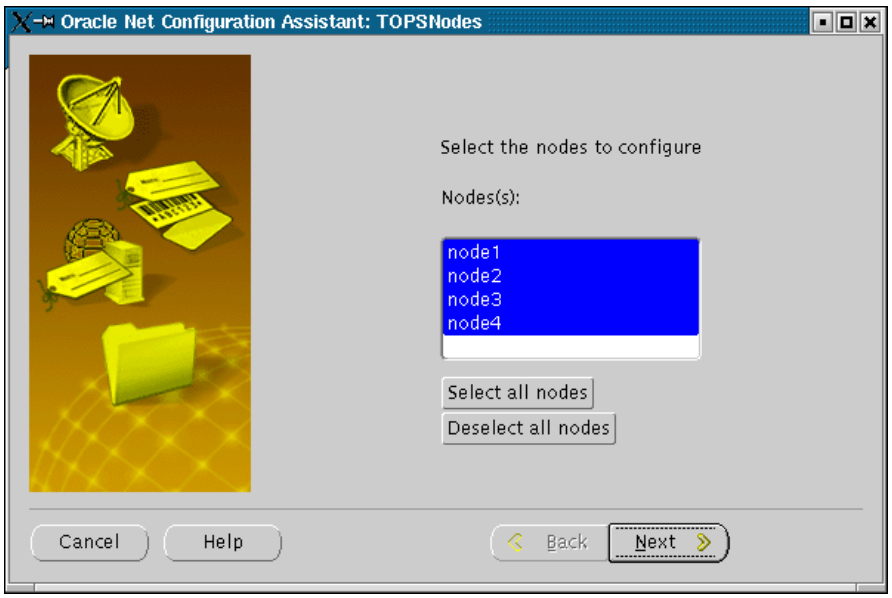


Figure 4-22 Node selection window

Click **Select all Nodes** and then **Next** to display the Oracle Net Configuration Welcome window.

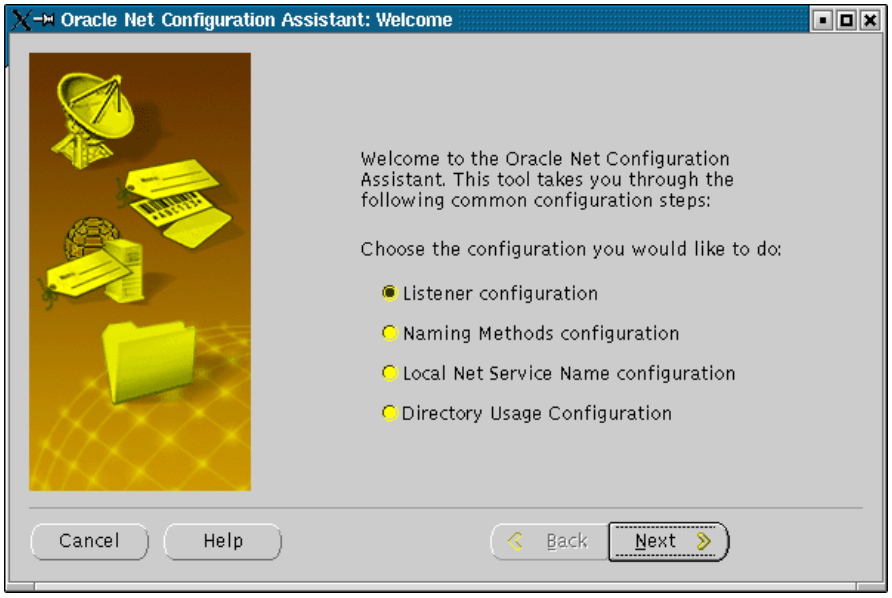


Figure 4-23 Welcome window

Select **Listener configuration** and then **Next**.

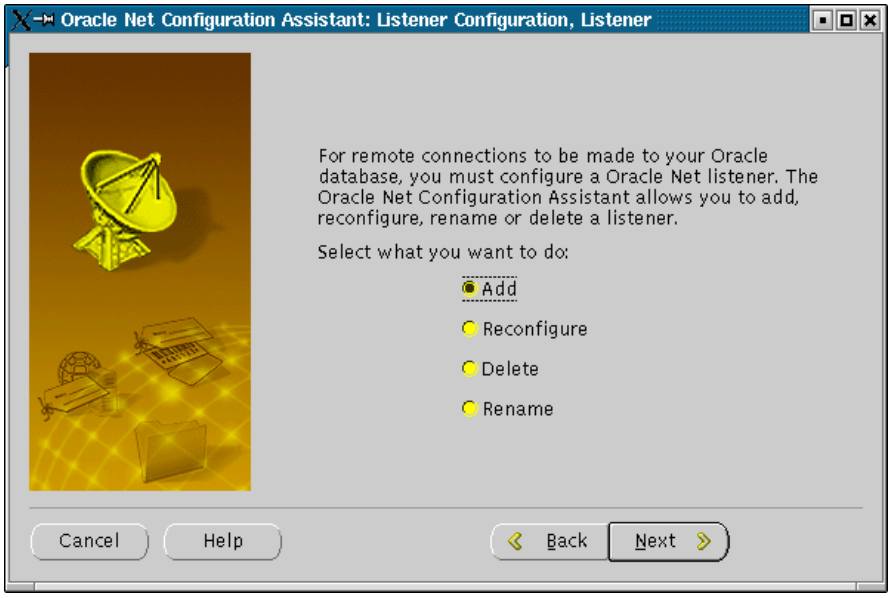


Figure 4-24 Listener configuration

Select **Add** and click **Next** to display the *Listener name* window, as shown in Figure 4-25, “Listener Name” on page 107.

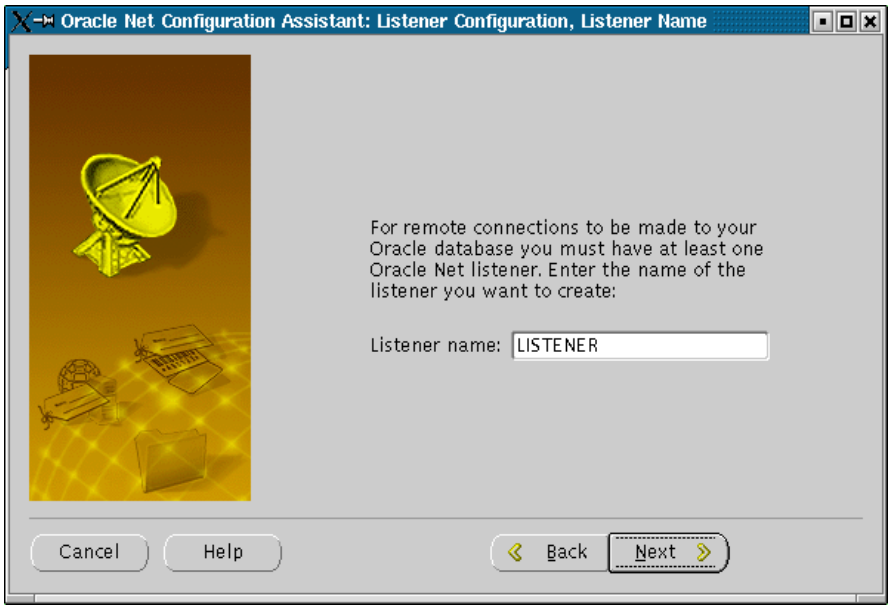


Figure 4-25 Listener Name

Click **Next** to accept the default listener name. The Select Protocols window is displayed, as shown in Figure 4-26 on page 107.

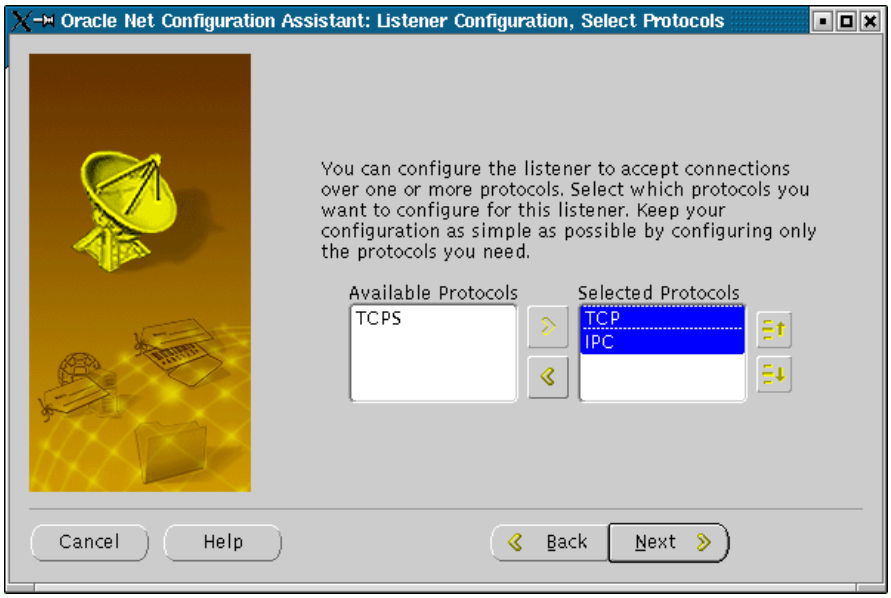


Figure 4-26 Select Protocol

Make sure TCP is selected. Click **Next** to continue and display the TCP/IP window (Figure 4-27).

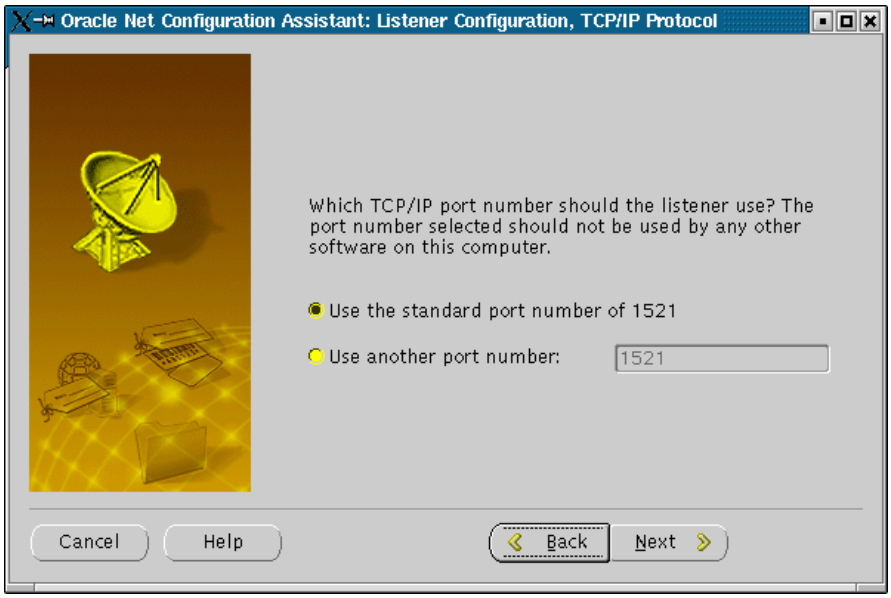


Figure 4-27 Select the TCP port window

Select the standard port number (1521) or enter another port number and click **Next**.

The following window asks you to configure another listener. We did not configure another listener in our test environment. Click **Next** to complete the listener configuration.

The $ORACLE_HOME/network/admin/listener.ora file contains all the configuration information needed to identify the addresses used by the server to accept client connections.

Click **Next** to return to the Welcome window, as shown in Figure 4-23 on page 106.

Database client network configuration (tnsnames.ora)

In this step we configure the database client connection identification, located in the $ORACLE_HOME/network/admin/tnsname.ora file. This is needed when local clients (such as sqlplus) need to connect to database services.

“Local Net Service name configurator” helps us to configure Oracle net service names.

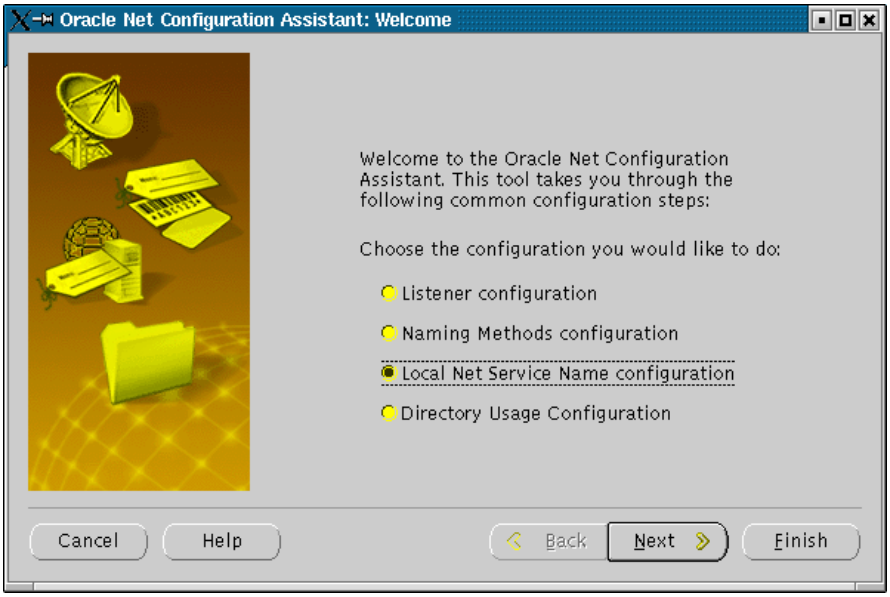


Figure 4-28 Select the Local Net Service Name configuration

Select **Local Net Service Name configuration** and click **Next** (see Figure 4-28 and Figure 4-29).

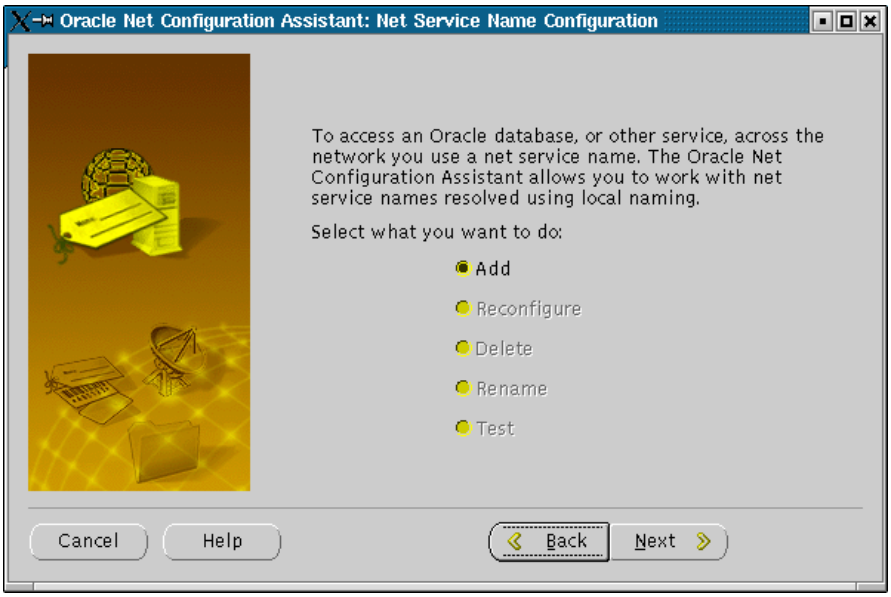


Figure 4-29 Add Net Service Name

Select **Add** and click **Next** to select the Oracle database version you want to connect to; see Figure 4-30 on page 110.

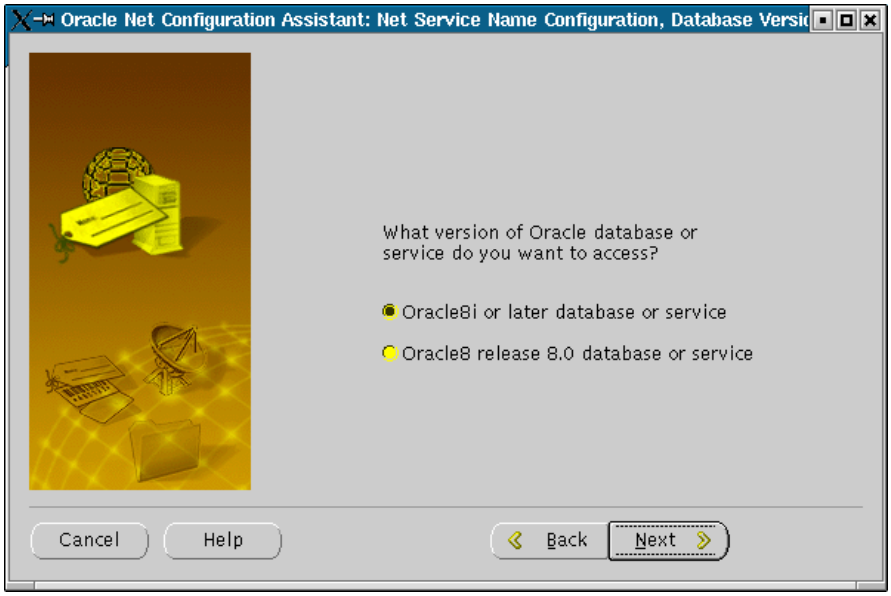


Figure 4-30 Oracle version

Choose **Oracle8i or later database or service** and click **Next** to continue, as shown in Figure 4-31, then type in the name of the service and click **Next** to display the Select Protocol window.

Figure 4-31 Specify database service name

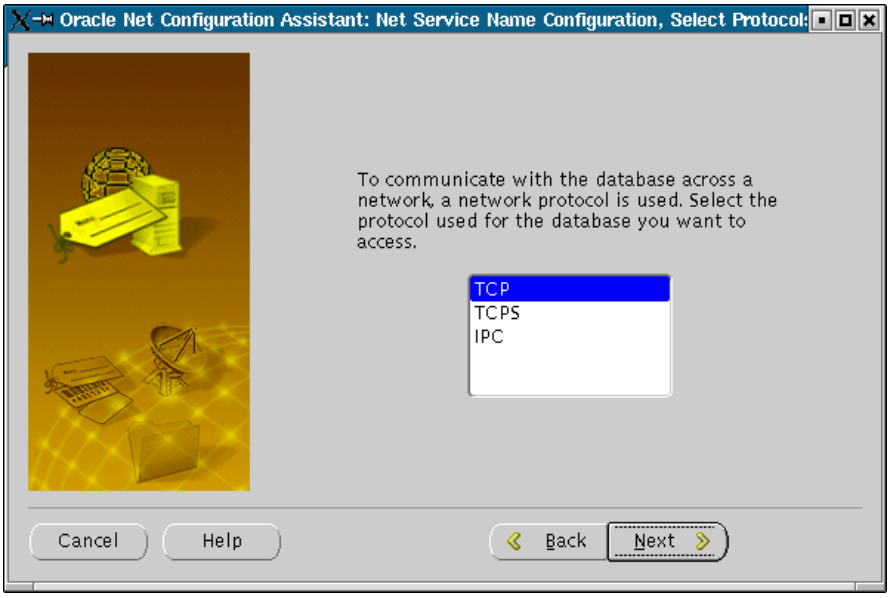


Figure 4-32 Select a protocol

There is a choice of three protocols. We chose **TCP**. Click **Next** to select the node (Host) name (or IP address).

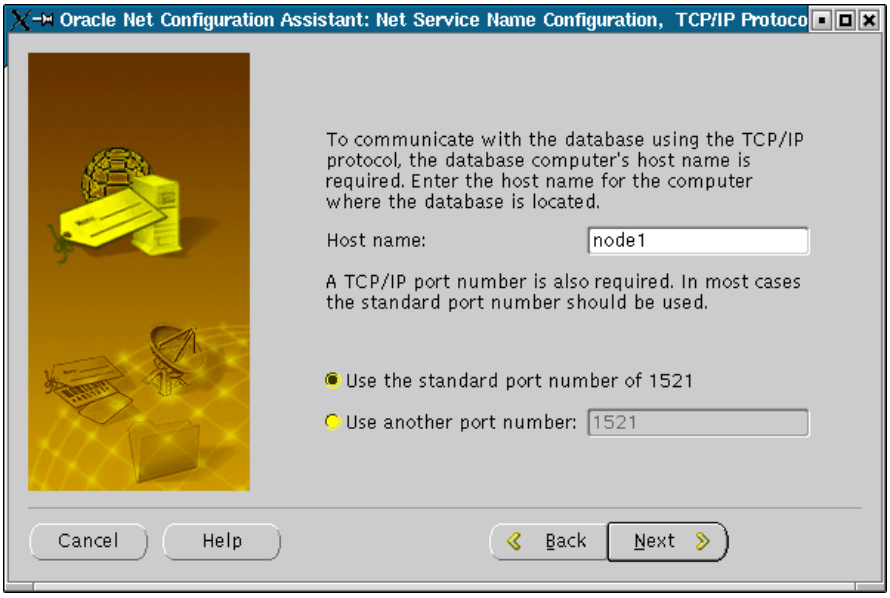


Figure 4-33 Node name selection window

Use the standard port number if you know that your database listener is configured to listen on port 1521, or, if different, use the port number specified during the listener configuration.

Click **Next** to display the Test window; see Figure 4-34 on page 112.

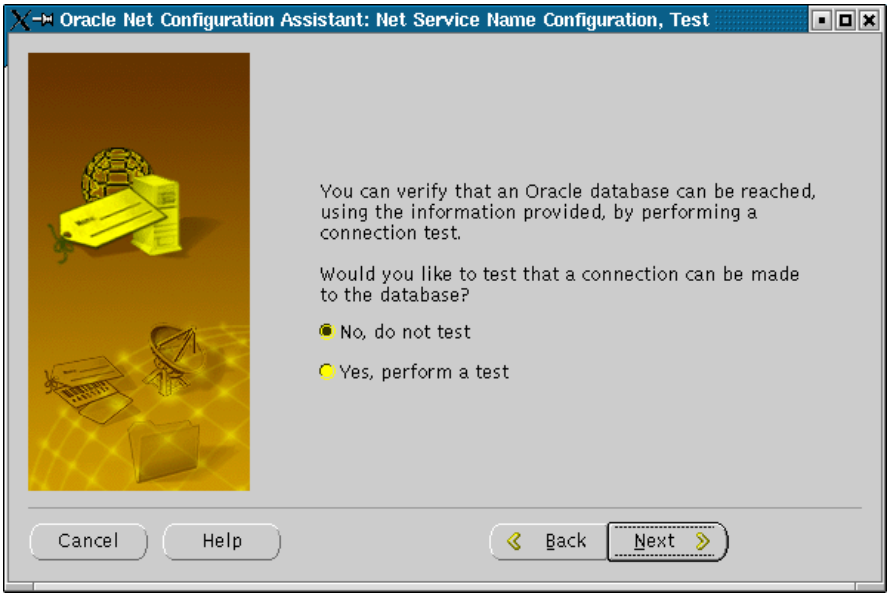


Figure 4-34 Connection test window

Select **No**, then click **Next** to display the Net Service Name window; see Figure 4-35.

This selection is due to the fact that a database and the corresponding listener are not yet active. The database service will be available after database creation and configuration, described later in this chapter.

Figure 4-35 Net Service Name

Enter the Net Service Name and click **Next**. Select **Yes** to add the entries for the other nodes. Our listener identifications are named LISTENER_RAC1 for node1, LISTENER_RAC2 for node2, and so on.

Note: It is important to select specific names, such as "LISTENER_<instance_NAME><instance_NUMBER>", where <instance_NUMBER> corresponds to the Oracle9*i* RAC thread ID, since this naming convention will be used by **dbca**, and the database creation process may fail if the naming convention is not followed.

4.3 Creating and validating the database

This section describes how to plan and create a database and the associated storage.

We created a Real Application Clusters Database using the **dbca** graphical user interface (GUI).

The database can also be created manually, using scripts. This is described later, in 4.3.6, "Manual creation of an Oracle9i RAC database" on page 129.

4.3.1 Database storage planning

For better planning and manageability, we recommend that you do not store the Oracle data files (database files, control files, and redo log files) in the $ORACLE_HOME directory. We created a separate GPFS file system named /data, shared by our four cluster nodes for storing our data files.

For storage planning and GPFS configuration, refer to 3.8, "ESS Configuration" on page 40 and 3.10.2, "GPFS cluster configuration" on page 52. Make sure that the file systems are mounted on all nodes and that user oracle has ownership and read/write permissions.

Ensure that there is enough space for the planned database. For our test environment, we chose the database file sizes shown in Table 4-2.

Table 4-2 Database storage

Database file	Size recommended	Operating system file name
SYSTEM	400 MB	/data/rac/system01.dbf
USERS	200 MB	/data/rac/users01.dbf
TOOLS	50 MB	/data/rac/tools01.dbf
TEMP	1 GB	/data/ractemp01.dbf
Control file1	default	/data/rac/control01.dbf
Control file2	default	/data/rac/control02.dbf
INDEX	3 GB	/data/rac/index01.dbf
Redo log Thread 1 Group1	20 MB	/data/rac/redo1_1.log
Redo log Thread 1 Group2	20 MB	/data/rac/redo2_1.log
Redo log Thread 1 Group3	20 MB	/data/rac/redo3_1.log
Redo log Thread 1 Grou4p	20 MB	/data/rac/redo4_1.log
UNDO1 Tablespace thread 1	1 GB	/data/rac/undotbs01.dbf
UNDO2 Tablespace thread 1	1 GB	/data/rac/undotbs02.dbf
UNDO3 Tablespace thread 1	1 GB	/data/rac/undotbs03.dbf

Database file	Size recommended	Operating system file name
UNDO4 Tablespace thread 1	1 GB	/data/rac/undotbs04.dbf
DATA_USER1	10GB	/data/rac/data01.dbf
DATA_USER2	10GB	/data/rac/data02.dbf
DATA_USER3	10GB	/data/rac/data03.dbf

4.3.2 DBCA configuration file creation

The Database Configuration Assistant uses a shared configuration file. This file must be on the shared space (GPFS file system); see Figure 4-11 on page 98.

The Oracle Universal Installer (OUI) initializes this file during the installation process. If this initialization does not complete successfully, you can manually initialize the configuration file.

The DBCA graphical utility needs to find the GSD (Global Services Daemon) running on nodes. The Server Control utility (the **srvctl** command) also sends requests to the GSD subsystem to execute administrative tasks on all instances in the cluster. The Server Control utility (**srvctl**) is a component of the Server Management utility (SRVM).

SRVM is included in Real Application Cluster and represents an integrated system management solution. It provides facilities for DBCA, OUI, and the Network Configuration Assistant (Netca) in order to operate with these tools on multiple nodes from one single point of control.

srvConfig.loc file

Check the location of the srvConfig.loc file. This file specifies the location of the RAC shared configuration file (needed by GSD and SRVCTL). Also, the HACMP cluster must be running (see Example 4-1 on page 86).

Verify the content of the srvConfig.loc file, as shown in Example 4-10. If this file does not exist, create it manually (using **vi**).

Example 4-10 The srvConfig.loc file

```
{node1:oracle}/var/opt/oracle-> cat srvConfig.loc
srvconfig_loc=/oracle/itso_rac <----This is the shared configuration file name
```

If the shared file (used by GSD) does not exist, perform the steps shown in Example 4-11.

Example 4-11 The shared parameter file initialization

```
{node1:oracle}->touch /oracle/itso_rac
{node1:oracle}->srvconfig -init <---- initialize the cluster wide SRVM configuration
{node1:oracle}/-> gsdclt start <----  This allow the SRVCTL utility to access the shared
                                      configuration information
```

4.3.3 Database creation using the Database Configuration Assistant

Oracle recommends the DBCA tool to create the RAC database. DBCA has templates for predefined environments that have all the Oracle9*i* RAC features.

Launch DBCA

To start DBCA, log in as user oracle and type at the command prompt:

```
{node1:oracle}/-> dbca -datafileDestination /data
```

The “-datafileDestination” option of **dbca** allows to specify a destination directory for the data files (valid only for the current **dbca** session). This option is convenient for space management and to avoid some typing errors in the DBCA dialogue windows.

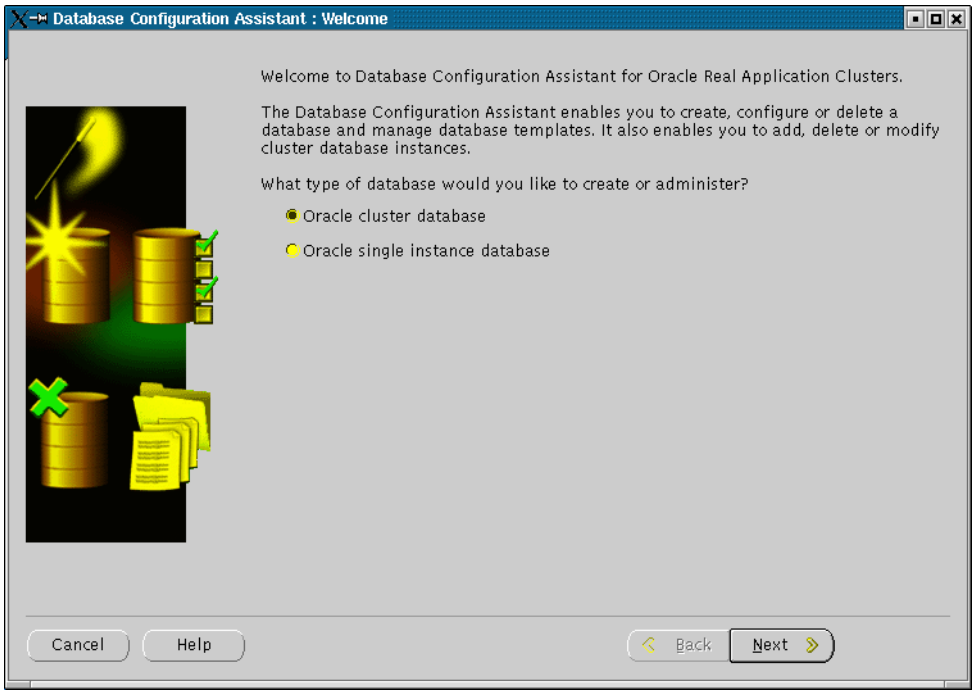


Figure 4-36 DBCA Welcome window

Choose **Create a cluster database** and click **Next**.

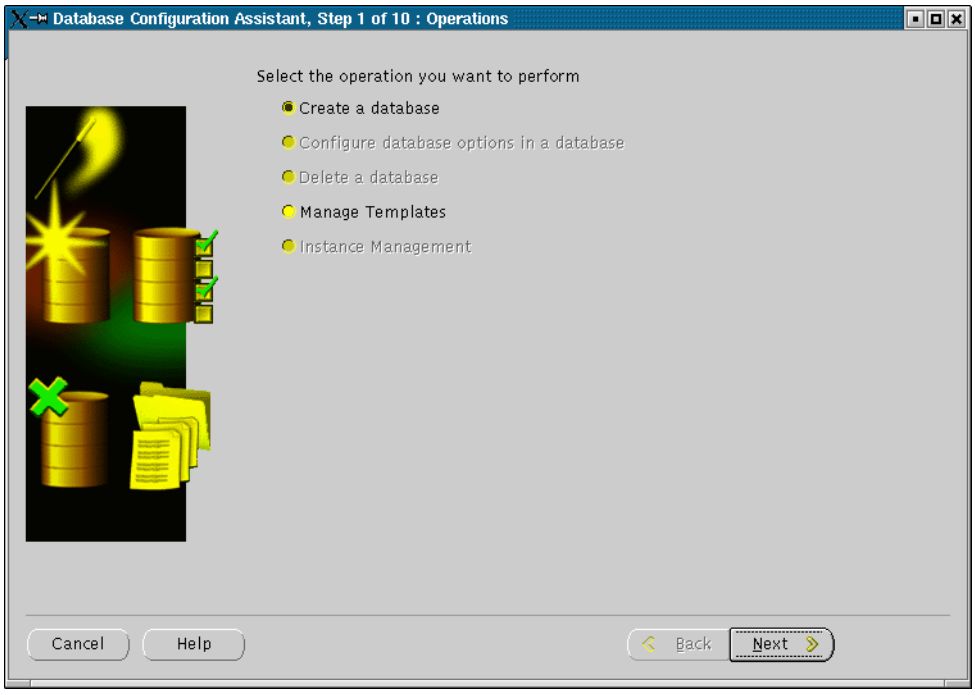


Figure 4-37 Create a database window

In the Node Selection window (Figure 4-38), select the cluster nodes. The local node is always selected.

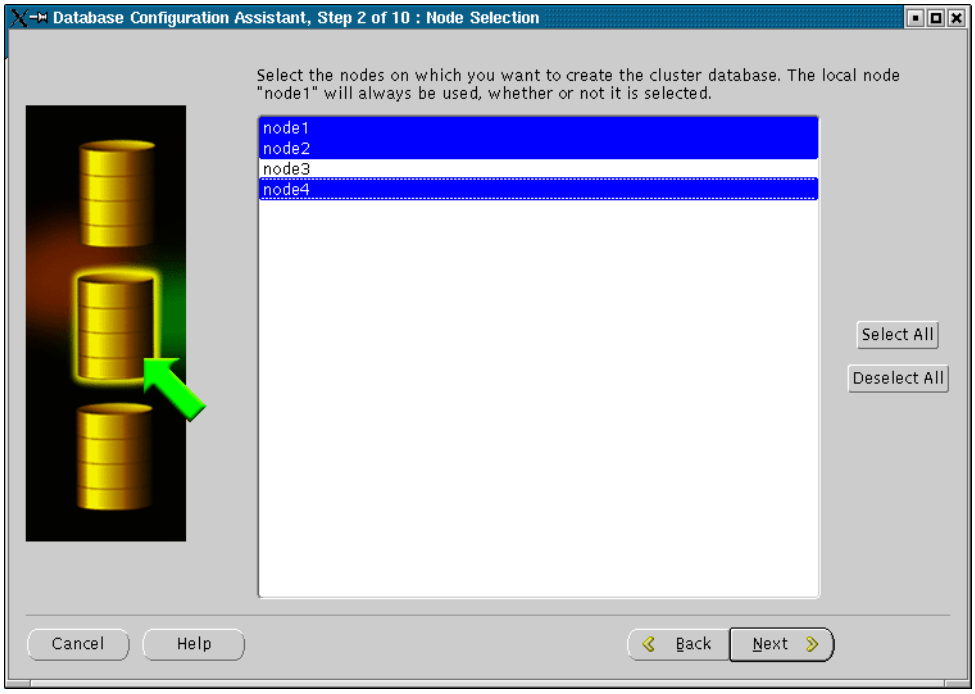


Figure 4-38 Node selection window

In the Database Templates window (shown in Figure 4-39), select **New Database**, then click **Next.** This lets you customize the parameters according to your needs.

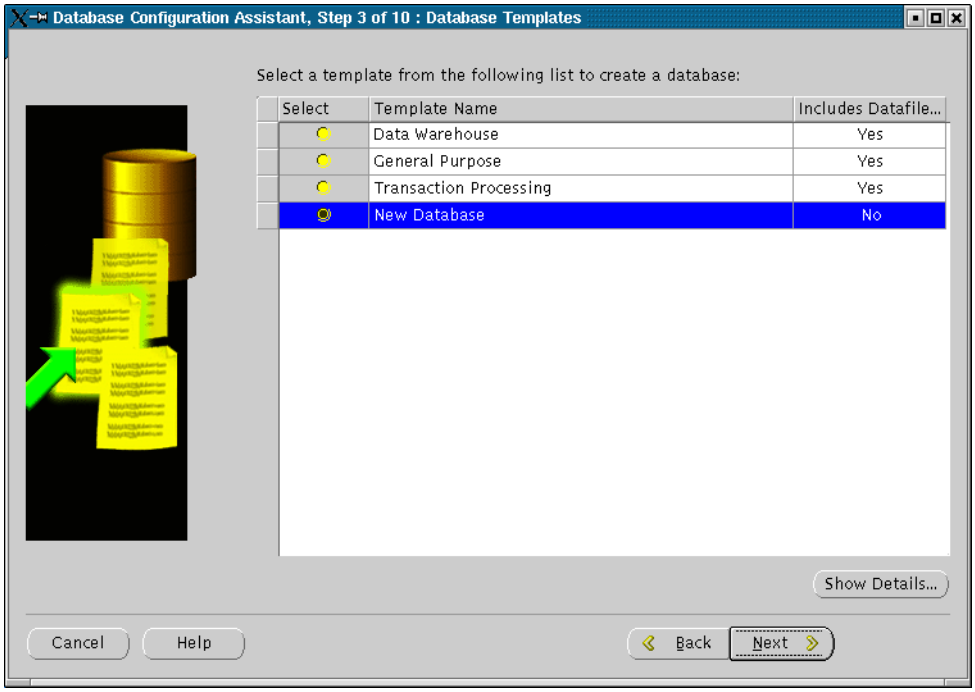


Figure 4-39 Database Templates window

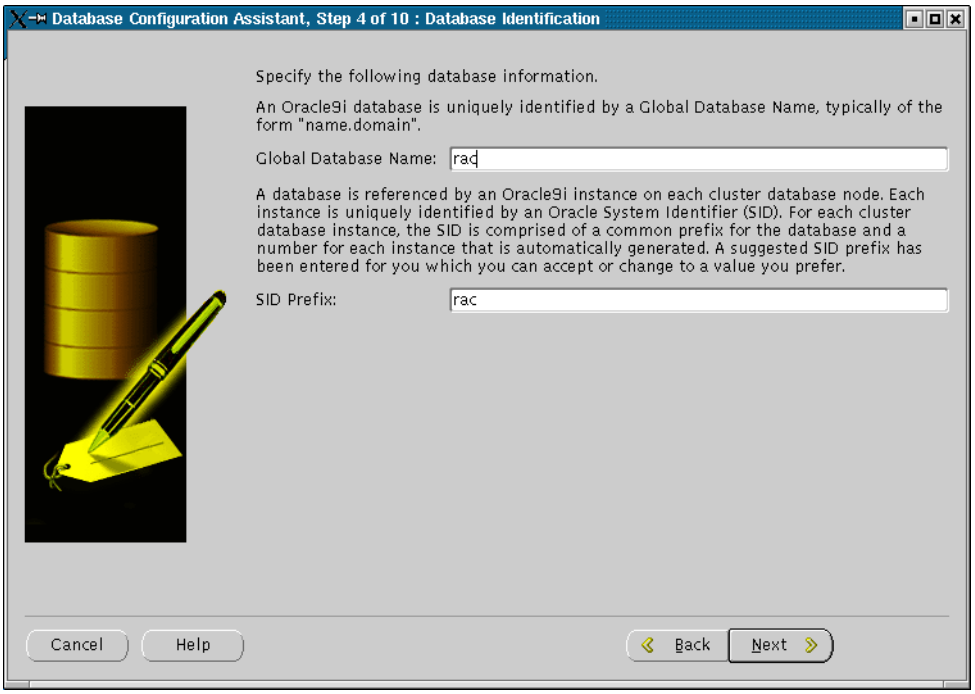


Figure 4-40 Database Identification

The Database Identification window appears. Type in the database name and the global sid name. The database name uniquely identifies the database. The DBCA tool names the instances by adding a thread ID to the global database name for each node in the cluster.

Click **Next** to display the Network Configuration window for this database, as shown in Figure 4-41.

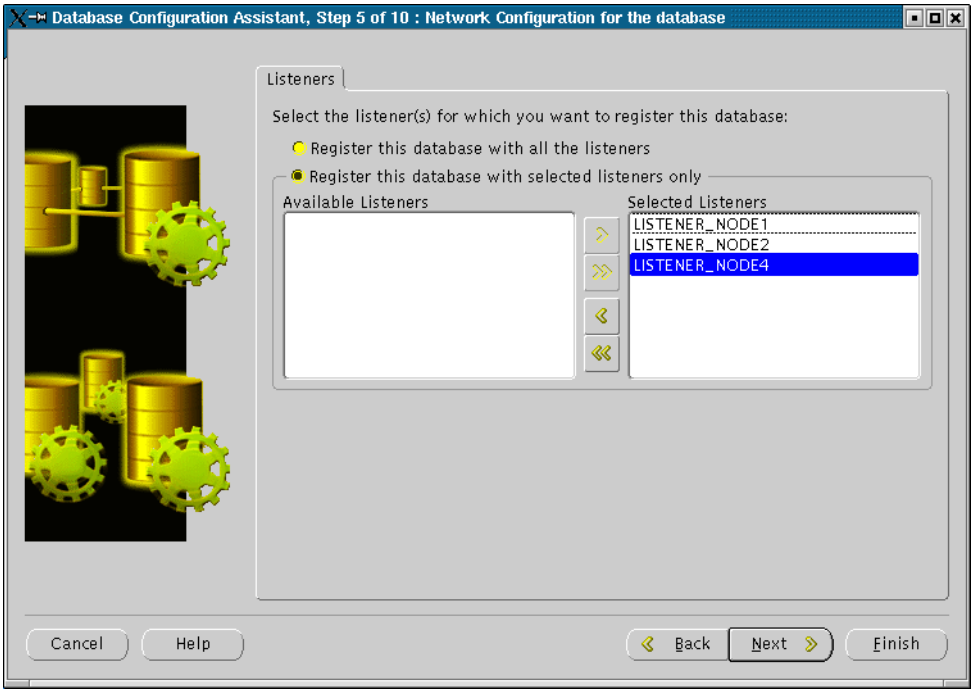


Figure 4-41 Network Configuration for the database

The DBCA tool configures the information for this Oracle9*i* database in the $ORACLE_HOME/network/admin/listener.ora file. It detects the cluster configuration and displays all the local listeners for the cluster nodes. Select all listeners with which you want to register this database.

The Oracle9*i* Release 2 (9.2) database service automatically registers its information (such as *service name* or *instance name)* with the default listener, named LISTENER. This feature does not require the listener.ora file.

Click **Next** to display the Database Feature window and selected features. Deselect the features that you don't need.

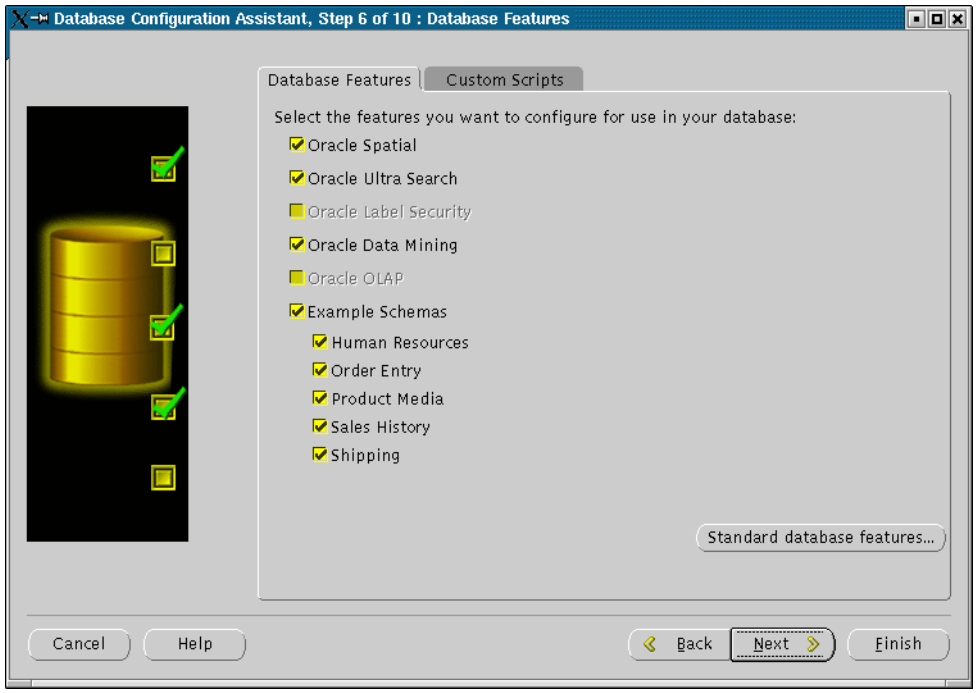


Figure 4-42 Database Features

Click **Next** and the File Location Variables window appears, as shown in Figure 4-43.

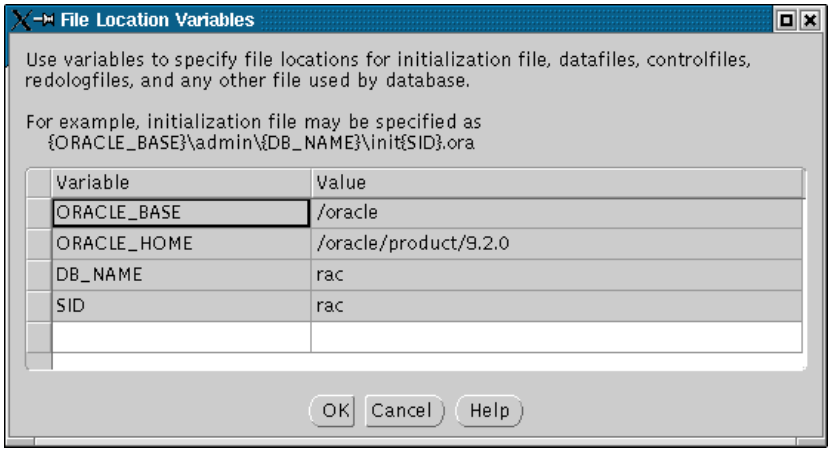


Figure 4-43 File Location Variables window

Click **Ok** to accept the settings and the Database Connection Options window appears; see Figure 4-44.

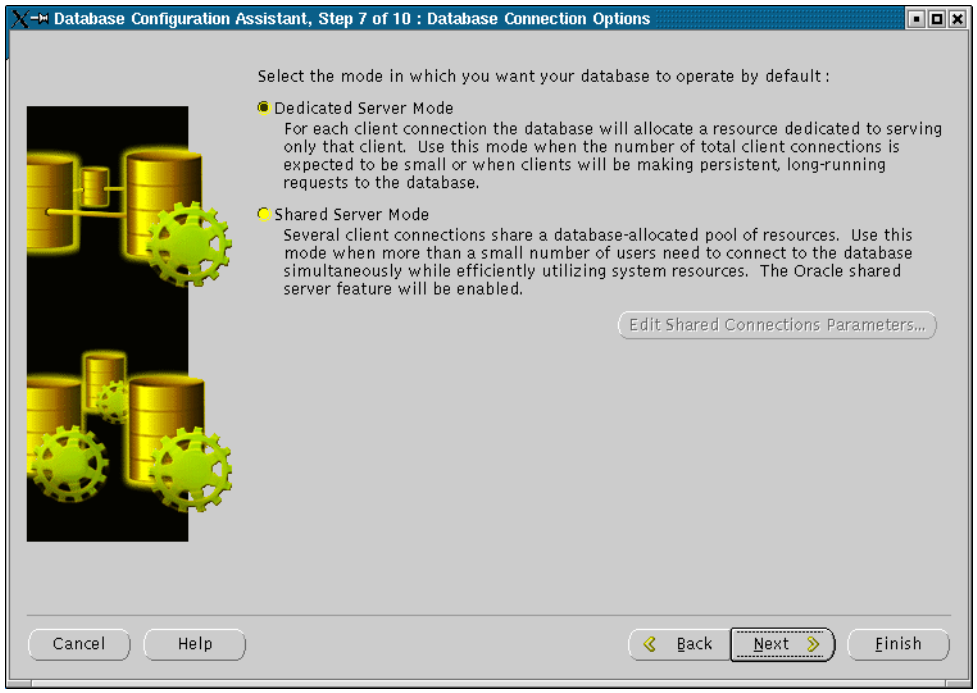


Figure 4-44 Database Connection Options

Dedicated Server Mode means that for each client process the server will dispatch a server communication process. This configuration is usually recommended when you want (and can afford) a dedicated server process for each incoming client connection (always consider the number of processes per CPU you plan to allow).

Shared Server Mode means that multiple client connections are allowed for one server process. This is useful in an environment with lots of clients so the systems can use the CPU and memory resources efficiently.

We selected **Dedicated Server Mode**. Next, the Initialization Parameters window is displayed (Figure 4-45 on page 120). Select the **DB Sizing** tab to change the DB block size to 16 KB, according to the GPFS block size (16 KB, as well).

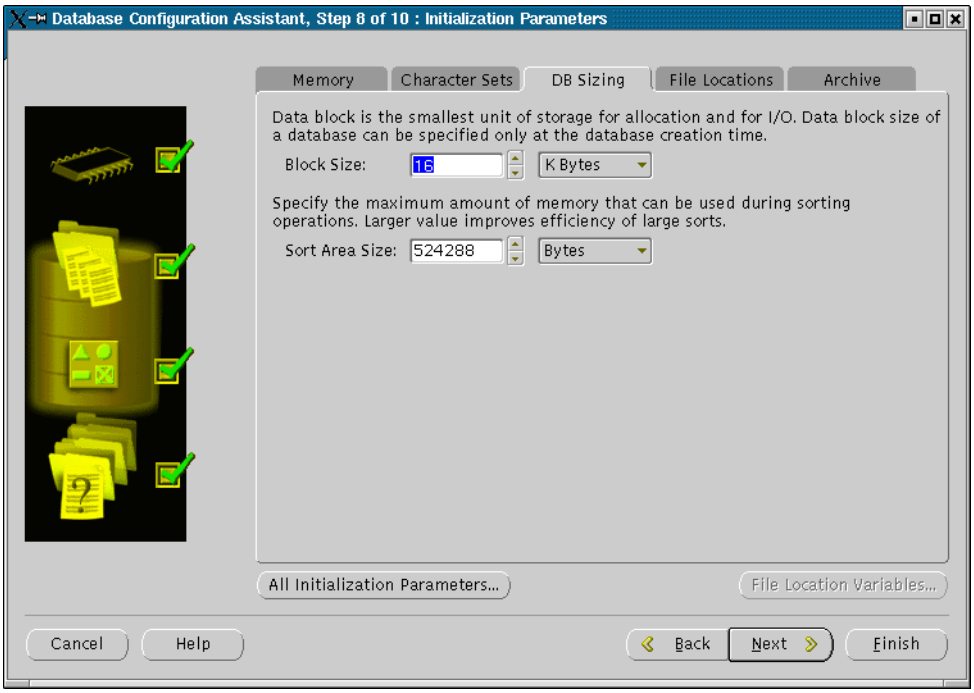


Figure 4-45 Initialization Parameters

We did not want to create the default instance parameter file, $ORACLE_HOME/network/admin/spfile, so we deselected this option, as shown in Figure 4-46. This file can be created later, after database creation, from the ASCII parameters file (init.ora).

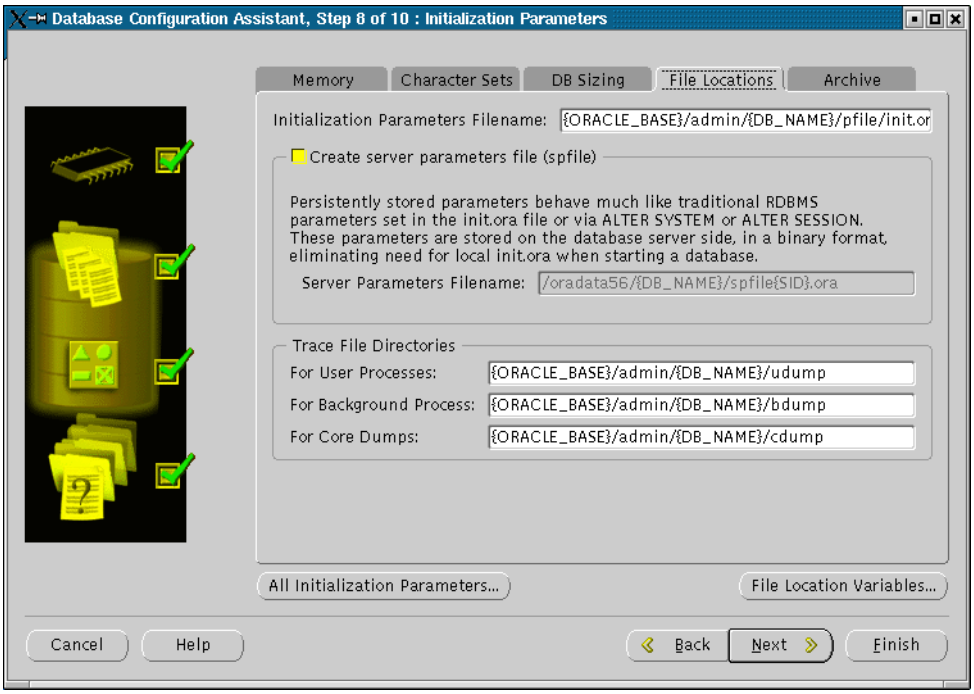


Figure 4-46 Parameter file window

We left other parameters with the default values, since we could change them later. You can check the parameters on All Initialization Parameters. Select **Next** and the Database Storage window appears (see Figure 4-47).

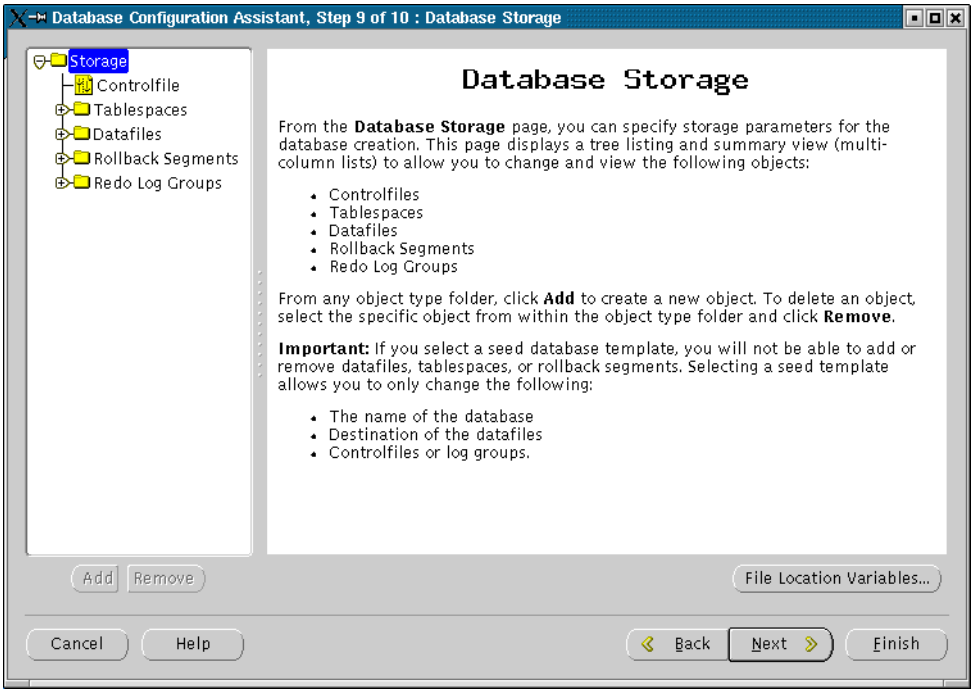


Figure 4-47 Database Storage window

Click the items under Datafiles,Tablespaces, Redo Log Groups to expand them and change their sizes, according to the planned database (see Table 4-2 on page 113).

Note: For consistency we changed the names of the redo log files to:

redo1_1.log
redo1_2.log
redo2_1.log
redo2_2.log
redo3_1.log
redo3_2.log
redo4_1.log
redo4_2.log.

Carefully check the values entered and click **Next** to display the Creation Options window, as shown in Figure 4-48 on page 122.

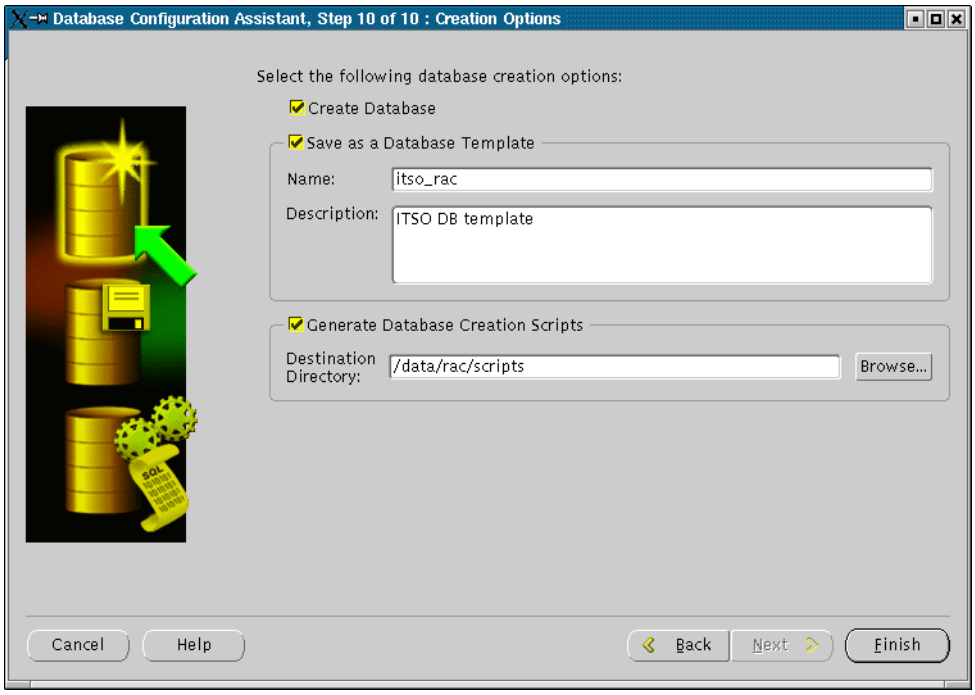


Figure 4-48 Creation Options window

You can also save all the options and selections from previous steps into a script and into an HTML template file.

At the end of this step, DBCA displays the pop-up window shown in Figure 4-49:

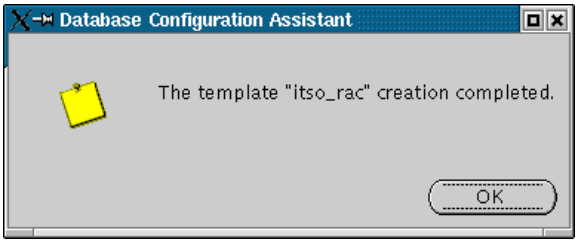


Figure 4-49 Template creation

Click **Ok** -> **Finish** to see all submitted parameters for database creation (Figure 4-50 on page 123). If everything is correct—this is the last chance to return—proceed to the database creation by again clicking **OK** -> **Finish**.

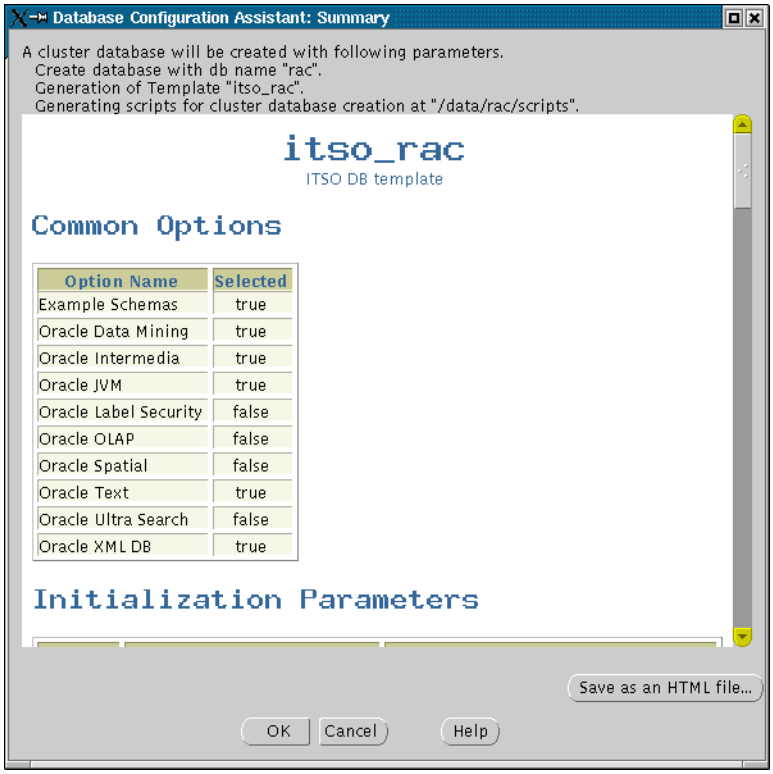


Figure 4-50 Summary window

Note: The listener{SID} is resolved to a listener address through a naming method into tnsnames.ora file. When you run DBCA with the dedicated server option and choose remote_listener, you must also set up the local_listener parameter.

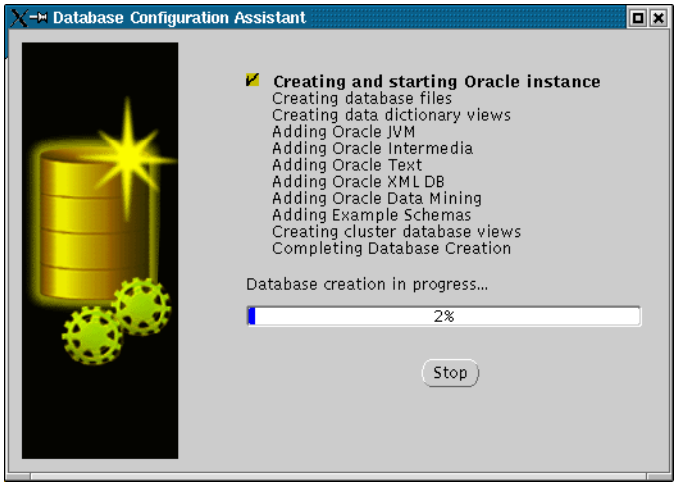


Figure 4-51 Database creating

At the end of the database creation (if successful) a window appears, asking you to change the password for sys and system DBA users (see Figure 4-52 on page 124).

Figure 4-52 Password window

Type in the passwords and click **Exit**.

To see the status of database creation you can check the alert log in the $ORACLE_BASE/admin/{DB_NAME}/udump directory. To check progress and see alert messages, type `tail -f alert_rac.log`.

This step concludes the database creation process.

4.3.4 Post database creation steps

Startup/shutdown of the database

Startup/shutdown operations can be performed either individually, on each node and instance separately, or from a single node, using the Oracle9*i* RAC tools provided.

The Server Manager (SRVM) utility provides a single point of control for administration tasks such as startup, stop, and status query of the database and its instances.

Also, with the Server Control (SRVCTL) utility, we can set up database and instance configurations in the cluster context. Other tools, such as Oracle Enterprise Manager (OEM) use SRVCTL to discover and monitor the cluster database nodes.

For more information, see the chapter “Administering Real Application Clusters Databases with the Server Control Utility” in the Oracle product manual *“Oracle9i Real Application Clusters Administration Release 2(9.2)”.*

The GSD daemon must be configured and running before using the SRVCTL utility.

- First, verify that the gsdctl daemon is started on all nodes in the cluster, by using on each node):

```
ps -ef |grep gsd
```

or

```
{node1:oracle}/oracle/home-> gsdctl stat
GSD is running on the local node
```

- If the daemon is started on all nodes, you can configure SRVCTL by adding the database (for example, on node1):

  ```
  {node1:oracle}/oracle-> srvctl add database -d rac  -o /oracle/product/9.2.0
  ```

  where `rac` is the name of the database and `/oracle/product/9.2.0` is the ORACLE_HOME.

- On the same node (node1), for each instance, execute the following command:

  ```
  {node1:oracle}/oracle-> srvctl add instance -d rac -i rac1 -n node1
  ```

- To perform administrative tasks, execute the commands shown in Example 4-12.

Example 4-12 Administrative tasks with SRVCTL

```
{node1:oracle}/oracle->  srvctl config <<< Verify the configuration>>>
rac
{node1:oracle}/oracle-> srvctl status database -d rac
Instance rac1 is running on node node1
Instance rac2 is running on node node2
Instance rac3 is running on node node3
Instance rac4 is running on node node4

node1:oracle}/oracle/home-> srvctl config -p rac -n node1
node1 rac1 /oracle/product/9.2.0 <<< repeat the command for configuration on each node>>>

<<<< We start the RAC instance on all the nodes once >>>
{node1:oracle}/-> srvctl start database -d rac

<<< We stop the RAC instance on all the nodes once >>>
{node1:oracle}/-> srvctl stop database -d rac

<<< We start only one instance into RAC >>>
{node1:oracle}/oracle/home-> srvctl start instance -d rac -i rac2

<<< We stop the instance into RAC >>>
{node1:oracle}/oracle/home-> srvctl stop instance -d rac -i rac2
```

spfile creation

Our default location for the database instances initialization files is $ORACLE_HOME/dbs.

For each instance we need to customize the PFILE static parameter file init{SID}.ora. At database creation time we chose *not* to create the instances dynamic parameter initialization file (spfile). Later on, if you decide to create this file, you can do this by using the ASCII parameter file, as shown in Example 4-13. This allows for dynamic parameter configuration for each instance separately.

Example 4-13 Creating the parameter file

```
$ sqlplus '/ as sysdba'
SQL> create spfile=/oracle/product/9.2.0.3/dbs/spfilerac1.ora from
pfile=/oracle/product/9.2.0.3/dbs/initrac1.ora
```

If you let the installer create the default spfile file, it creates only a server parameter file, spfile.ora, which is the same for all instances.

The instance startup profile lookup order is (where the instance looks for the initialization parameters):

1. $ORACLE_HOME/dbs/spfile*sid*.ora, if it exists
2. $ORACLE_HOME/dbs/spfile.ora
3. $ORACLE_HOME/dbs/init*sid*.ora

If any error occurs during the instance startup, check the instance alert log file (see the $ORACLE_BASE/admin/{DB_NAME}/udump directory).

4.3.5 Oracle Net Services configuration for RAC

listener.ora file configuration

If you choose to use the listener.ora file, then it would have been configured previously, when you ran **netca** and **dbca**.

This file, located in the $ORACLE_HOME/network/admin directory, must be checked before launching the database. The listener stanza for RAC1 must be similar to the one shown in Example 4-14.

Example 4-14 listener.ora file

```
LISTENER_NODE1 =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS_LIST =
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = TCP)(HOST = node1)(PORT = 1521))
      )
    )
  )

SID_LIST_LISTENER_NODE1 =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = /oracle/product/9.2.0)
      (PROGRAM = extproc)
    )
    (SID_DESC =
      (ORACLE_HOME = /oracle/product/9.2.0)
      (SID_NAME = rac1)
    )
    )
  )
```

tnsnames.ora file configuration

This file is also located in the $ORACLE_HOME/network/admin/ directory, and contains the information needed by the clients to connect to the database instances. The tnsnames.ora file we used in our environment is shown in Example 4-15.

Example 4-15 tnsnames.ora file configuration

```
# TNSNAMES.ORA Network Configuration File

# This entry does not specify an instance, but just the database.
# Because the load balancing is on, you will be connected to a different
# instance each time, in a round robin mode.
```

```
# When you use this entry, you don't know the instance you will be
# connected to. But the load is spread over all the instances.
#
# If the instance you are connected to fails, the Transparent Application
# Failover (TAF) will failover your session to another instance.

RAC =
  (DESCRIPTION =
    (ENABLE=BROKEN)
    (LOAD_BALANCE=ON)
    (FAILOVER=ON)
      (ADDRESS = (PROTOCOL = TCP)(HOST = node1)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node2)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node3)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node4)(PORT = 1521))

    (CONNECT_DATA =
      (SERVICE_NAME = RAC)
      (failover_mode=(type=select)(method=basic)(retries=20)(delay=5))
    )
  )


# The following entries tries to connect to the first instance in the list.
# Only if this instance is down, the second instance is tried, and so on.
# There is no load balance. Assuming that the first instance is up,
# you will be connected all the time to this instance.
#
# If the instance you are connected to fails, the Transparent Application
# Failover (TAF) will failover your session to the next instance in the list.

RAC1 =
  (DESCRIPTION =
    (ENABLE=BROKEN)
    (LOAD_BALANCE=OFF)
    (FAILOVER=ON)
      (ADDRESS = (PROTOCOL = TCP)(HOST = node1)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node2)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node3)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node4)(PORT = 1521))

    (CONNECT_DATA =
      (SERVICE_NAME = RAC)
      (failover_mode=(type=select)(method=basic)(retries=20)(delay=5))
    )
  )

RAC2 =
  (DESCRIPTION =
    (ENABLE=BROKEN)
    (LOAD_BALANCE=OFF)
    (FAILOVER=ON)
      (ADDRESS = (PROTOCOL = TCP)(HOST = node2)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node3)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node4)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node1)(PORT = 1521))

    (CONNECT_DATA =
      (SERVICE_NAME = RAC)
      (failover_mode=(type=select)(method=basic)(retries=20)(delay=5))
```

```
    )
  )

RAC3 =
  (DESCRIPTION =
    (ENABLE=BROKEN)
    (LOAD_BALANCE=OFF)
    (FAILOVER=ON)
      (ADDRESS = (PROTOCOL = TCP)(HOST = node3)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node4)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node1)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node2)(PORT = 1521))

    (CONNECT_DATA =
      (SERVICE_NAME = RAC)
      (failover_mode=(type=select)(method=basic)(delay=10)(retries=50))
    )
  )

RAC4 =
  (DESCRIPTION =
    (ENABLE=BROKEN)
    (LOAD_BALANCE=OFF)
    (FAILOVER=ON)
      (ADDRESS = (PROTOCOL = TCP)(HOST = node4)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node1)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node2)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node3)(PORT = 1521))

    (CONNECT_DATA =
      (SERVICE_NAME = RAC)
      (failover_mode=(type=select)(method=basic)(retries=20)(delay=5))
    )
  )
```

- The RAC stanza contains information for all the listeners in the cluster (each instance has its own listener). This information is used by instances to register the listeners at startup.
- The stanzas RAC1, RAC2, RAC3, and RAC4 contain connect information for instances located on node1, node2, node3, and node4.

In order to enable load balancing, set LOAD_BALANCING to `on` in the tnsnames.ora file for the RAC stanza. This parameter is set to `on` by default by **`netca`** or **`dbca`**.

For Transparent Application Failover (TAF), set the parameter FAILOVER to `on` and also add a FAILOVER_MODE in the CONNECT _DATA stanza.

The FAILOVER_MODE has two important parameters:

- TYPE - which specifies what actions are taken in case of failover: the lost client session will be automatically recreated, the select statements are preserved, and the insert/update/delete statements will be “undone”.
- METHOD - which shows how the failover is performed.

Note: On the AIX client machine, set the ENABLE parameter to BROKEN in the tnsnames.ora file. Verify and set the tcp_keepidle parameter (in the network options -no configuration) to 240 (no -o tcp_keepidle=240). This parameter represents the total number of half-seconds for a TCP connection to remain *alive*, in the *idle* state. The default value is 14400, which means 2 hours!

If you do not set this parameter on your AIX client machine, then if the Oracle instance on the server, or the listener, goes down, the client will wait for two hours (!) before attempting to reconnect to another instance (according to the tnsnames.ora file).

4.3.6 Manual creation of an Oracle9*i* RAC database

If you do not want to use the graphical tools (**dbca** and **netca**) to create a database, you can do this manually. This may sometimes be useful, but the steps must be strictly followed in order to get a sane database environment.

The environment setup has to be done the same way as in 4.2, “Oracle9i RAC installation and configuration (on GPFS)” on page 88. Also, the Oracle-related variables are the same.

After Oracle code installation (using OUI), verify the sqlplus client, using the **sqlplus ‘/as sysdba’** command.

Database creation is performed on a single node. When this operation is finished, you need to add the necessary information for the cluster database in the initialization files. Keep in mind that this procedure does not perform all the verifications that the graphic tool (**dbca**) performs, so you have to validate the cluster configuration, or the other cluster instances may fail to start.

Set up the following variables:

- Database Name: RAC
- SID prefix: RAC (for simplicity)
- ORACLE_SID is defined by adding the instance thread ID to the SID prefix. This must be set on all the nodes in the cluster.

On node1, rename the instance initialization file in the $ORACLE_HOME/dbs directory to initrac1.ora. We chose to edit the init.ora file and add the necessary stanzas for all instances. Example 4-16 shows the content of the initialization file for the first instance (rac1). Repeat for the other instances (rac2, rac3, and rac4) by creating one file for each instance, named init{SID}.ora.

We chose to edit the init.ora file in one step and add the following stanzas for each instance:

INSTANCE_NAME

INSTANCE_NUMBER

UNDO_TABLESPACE

THREAD

Example 4-16 initrac1.ora file

```
cluster_database=true
cluster_database_instances=4
db_name=RAC
db_domain=""

# first instance
```

```
RAC1.thread=1
RAC1.instance_name=RAC1
RAC1.instance_number=1
RAC1.undo_tablespace=UNDOTBS1

# second instance
RAC2.thread=2
RAC2.instance_name=RAC2
RAC2.instance_number=2
RAC2.undo_tablespace=UNDOTBS2

# third instance
RAC2.thread=3
RAC2.instance_name=RAC3
RAC2.instance_number=3
RAC2.undo_tablespace=UNDOTBS3

# fourth instance
RAC2.thread=4
RAC2.instance_name=RAC4
RAC2.instance_number=4
RAC2.undo_tablespace=UNDOTBS4

control_files=("/data/rac/control01", "/data/rac/control02", "/data/rac/control03")

db_block_size=16384
db_cache_size=52428800
open_cursors=300
timed_statistics=TRUE
compatible=9.2.0
java_pool_size=52428800
large_pool_size=1048576
shared_pool_size=52428800
processes=150
fast_start_mttr_target=300
resource_manager_plan=SYSTEM_PLAN
sort_area_size=524288
undo_management=AUTO
```

Prepare a database creation script, similar to the one in Example 4-17.

Example 4-17 Create.sql script

```
CREATE DATABASE RAC
CONTROLFILE REUSE
MAXINSTANCES 8
MAXLOGHISTORY 100
MAXLOGFILES 100
MAXLOGMEMBERS 5
MAXDATAFILES 200
DATAFILE '/data/rac/system01' SIZE 325M REUSE
UNDO TABLESPACE "UNDOTBS1" DATAFILE '/data/undotbs01' SIZE 200M REUSE
CHARACTER SET WE8ISO8859P1
NATIONAL CHARACTER SET AL16UTF16
LOGFILE GROUP 1 ('/data/redo01_1') SIZE 20M REUSE,
GROUP 2 ('/data/redo01_2') SIZE 20M REUSE;
```

Comments:

The MAXINSTANCE parameter is mandatory in our scripts and specifies how many instance will be part of this Oracle9*i* RAC installation.

The MAXLOGFILES parameter is chosen by using the formula:

MAXLOGFILES=number_of_log_files*number_of_threads*number_of_groups

Start database creation

We can start the database creation process by connecting on node1 as “oracle”, and executing the following:

Example 4-18 Launching the database creation script

```
{node2:oracle}/-> sqlplus '/ as sysdba'
SQL> startup nomount
SQL> @/oracle/admin/rac/create/createdb.sql
.....................
```

After this step, create the user tablespace, temporary tablespace, and tools tablespace, sized according to your needs.

Post database creation actions

Oracle provides the necessary scripts to configure a fresh database, for creating the database catalog, views, synonyms, etc.

The scripts are catalog.sql, catexp.sql, and catproc.sql. To create the cluster database information, run the catclust.sql script, which creates RAC-specific views and/or tables. Also, connect as user *system* and execute the publd.sql script.

When these operations are completed, the instance on node1 should be up and running. Use this instance to add the other instances to the cluster, as shown in Example 4-19. Repeat for instances rac3 and rac4.

Example 4-19 Adding the second instance into RAC

```
create undo tablespace UNDOTBS2 datafile '/data/undotbs02' SIZE 200M REUSE EXTENT
MANAGEMENT LOCAL;

alter database add logfile thread 2
GROUP 4 ('/data/redo2_1') SIZE 20M REUSE,
GROUP 5 ('/data/redo2_2') SIZE 20M REUSE;
alter database enable public thread 2;
```

The other instances can be now started on their respective nodes.

For more information, see the Oracle documentation *Oracle9i Real Application Clusters Setup and Configuration Release 2(9.2)* and *Oracle9i Database Administrator’s Guide Release 2.*

4.4 Oracle9i general tuning considerations on AIX platforms

This section describes some of the administration issues when running Oracle9*i* (RAC or non-RAC) on AIX. These guidelines are general recommendations for running Oracle9i on IBM pSeries/AIX platforms.

Note: This section has been added as a general reference, because it is not configuration specific.

The following topics are covered:

- Memory and paging
- Disk I/O issues
- CPU scheduling and process priorities

4.4.1 Memory and paging on JFS/JFS2 file systems

Memory contention occurs when processes require more memory than is physically available. To cope with the shortage, the system exchanges program and data memory pages between real memory (RAM) and disks (paging).

Controlling buffer-cache paging activity

Excessive paging activity decreases performance substantially. This can become a problem with database files created on journaled file systems (JFS and JFS2). In this situation, a large number of SGA data buffers might also have analogous file system buffers containing the most frequently referenced data.

The behavior of the AIX file buffer cache manager can have a significant impact on performance. It can cause an I/O bottleneck, resulting in lower overall system throughput.

Note: AIX 5L introduces implicit **Direct I/O support**, which allows database files to be on file systems while bypassing the operating system's buffer cache. For installations using file systems for storage of data files, Oracle Corporation recommends that you enable Direct I/O on file systems containing Oracle data files. This is accomplished by using the mount option **`dio`.**

GPFS file systems are always mounted with Direct I/O enabled, whether or not the **`dio`** option was specified.

On AIX, it's possible to tune the buffer-cache paging activity, but this must be done carefully. Use the **`vmstat`** command to tune the following AIX system parameters:

Table 4-3 Virtual Memory Parameters

Parameter	Description
MINFREE	The minimum free-list size. If the free-list space in the buffer falls below this size, the system uses page stealing to replenish the free list.
MAXFREE	The maximum free-list size. If the free-list space in the buffer exceeds this size, the system stops using page stealing to replenish the free list.
MINPERM	The minimum number of permanent buffer pages for file I/O.
MAXPERM	The maximum number of permanent buffer pages for file I/O.

For more information on AIX system parameters, see also:

AIX 5L Version 5.2: Performance Management Guide

Log on as the root user and use the **vmo** command to change these limits. The AIX **vmo** command is operating system version-specific. If you run the **vmtune** command from one release on a different AIX release, the operating system might fail.

Tuning the AIX file buffer cache

The purpose of the AIX file buffer cache is to reduce disk access frequency when journaled file systems are used. If this cache is too small, disk usage increases and potentially saturates one or more disks. If the cache is too large, memory is wasted.

For information on the implications of increasing the AIX file buffer cache, see also “Controlling buffer-cache paging activity” on page 132.

You can configure the AIX file buffer cache by adjusting the MINPERM and MAXPERM parameters. Generally, if the buffer hit ratio is low (less than 90 percent), as determined by the **sar -b** command, increasing the MINPERM parameter value might help. If maintaining a high buffer hit ratio is not critical, decreasing the MINPERM parameter value increases the physical memory available. Refer to your AIX documentation for more information on increasing the size of the AIX file buffer cache.

The performance gain cannot be quantified easily, because it depends on the degree of multiprogramming and the I/O characteristics of the workload.

Tuning the MINPERM and MAXPERM parameters

AIX provides a mechanism to loosely control the ratio of page frames used for files versus those used for computational (working or program text) segments, by adjusting the MINPERM and MAXPERM values according to the following guidelines:

- If the percentage of real memory occupied by file pages falls below the MINPERM value, the page-replacement algorithm steals both file and computational pages, regardless of repage rates.
- If the percentage of real memory occupied by file pages rises above the MAXPERM value, the page-replacement algorithm steals both file and computational pages.
- If the percentage of real memory occupied by file pages is between the MINPERM and MAXPERM parameter values, the virtual memory manager (VMM) normally steals only file pages, but if the repaging rate for file pages is higher then the repaging rate for computational pages, the computational pages are stolen as well.

Use the following algorithm to calculate the default values:

MINPERM (in pages) = ((number of page frames)-1024) * 0.2
MAXPERM (in pages) = ((number of page frames)-1024) * 0.8

Use the following command to change the value of the MINPERM parameter to 5 percent of the total number of page frames, and the value of the MAXPERM parameter to 20 percent of the total number of page frames:

```
# vmo -p 5 -P 20
```

The default values are 20 percent and 80 percent, respectively.

To optimize for quick response when opening new DB connections, adjust the MINFREE parameter in order to maintain enough free pages in the system to load the application into memory without adding additional pages to the free list. To determine the real memory size (resident set size, working set) of a process, use "**ps v <process id>**". Set MINFREE to this size or eight frames, whichever is larger.

If the database files are stored on RAW devices, or if Direct I/O is used, you can set the MINPERM and MAXPERM parameters to low values, for example 5 percent and 20 percent, respectively. This is because the AIX file buffer cache is *not* used for RAW devices *or* for Direct I/O. The memory might be used for other purposes, such as the Oracle System Global Area (SGA).

Controlling paging

Constant and excessive paging indicates that the real memory is over-committed. In general, you should:

- Avoid constant paging unless the system is equipped with very fast storage that makes paging between memory and expanded storage much faster than Oracle can read and write data between the SGA and disks.
- Allocate limited memory resources to where it is most beneficial to system performance. It is sometimes a recursive process of balancing the memory resource requirements and trade-offs.

If memory is not adequate, build a prioritized list of memory-requiring processes and elements of the system. Assign memory to where the performance gains are the greatest. A prioritized list might look as follows:

1. OS and RDBMS kernels
2. User and application processes
3. Redo log buffer
4. PGAs and shared pool
5. Database block buffer caches

For instance, if you query Oracle dynamic performance tables and views, and find that both the shared pool and database buffer cache require more memory, assigning the limited spare memory to the shared pool might be better than assigning it to the database block buffer caches.

The following AIX commands provide paging status and statistics:

```
vmstat -s
vmstat interval [repeats]
sar -r interval [repeats]
```

Tuning the log archive buffers

You may be able to improve the speed of the database archiving operation, particularly if transactions are long or numerous, by increasing the LOG_BUFFER size. Monitor the log file I/O activity and system throughput to find the optimum LOG_BUFFER size. Tune this parameter carefully so that the overall performance of normal database activity is not degraded.

Note: The parameter LOG_ARCHIVE_BUFFER_SIZE was made obsolete in Oracle8i.

I/O buffers and SQL*Loader

For high-speed data loading, such as when using the SQL*Loader direct path option, in addition to loading data in parallel, the CPU spends most of its time waiting for I/O to complete. By increasing the number of buffers, you can usually push CPU usage harder, thereby increasing overall throughput.

The number of buffers you choose (set by the SQL*Loader BUFFERS parameter) depends on the amount of available memory and how hard you want to push CPU usage. The

performance gains depend on CPU usage and the degree of parallelism that you use when loading data.

For more information on the SQL*Loader utility, see *Oracle9i Database Utilities Release 2 (9.2) Part No. A96652-01.*

BUFFER parameter for the import utility

The BUFFER parameter for the Import utility should be set to a large value to optimize the performance of high-speed networks when they are used. For instance, if you use the IBM RS/6000 Scalable POWERparallel® Systems (SP) switch, you should use a value of at least 1 MB.

4.4.2 AIX Logical Volume Manager

The AIX Logical Volume Manager (LVM) can stripe data across multiple disks to reduce disk contention. The primary objective of striping is to achieve high performance when reading and writing large sequential files. Effective use of the striping features in the LVM allows you to spread I/O more evenly across disks, resulting in greater overall performance.

Note: If you are using Oracle Service Manager, you should not use LVM for striping. OSM performs this function.

Design a striped logical volume

When you define a striped logical volume, specify the items listed in Table 4-4.

Table 4-4 LVM parameters

Item	Recommended Settings
Drives	At least two physical drives. The drives should have minimal activity when performance-critical sequential I/O is executed. Sometimes you might need to stripe the logical volume between two or more adapters.
Stripe unit size	Although the stripe unit size can be any power of 2 from 2 KB to 128 KB, stripe sizes of 32 KB and 64 KB are good values for most workloads. For Oracle database files, the stripe size must be a multiple of the database block size.
Size	The number of physical partitions allocated to the logical volume must be a multiple of the number of disk drives used.
Attributes	Cannot be mirrored. Set the copies attribute to a value of 1.

Suggested Striped Logical Volume parameters

Note: If using the logical volume as a RAW device or as a file system mounted with Direct I/O, the Oracle database will perform read-ahead itself.

On the contrary, if using a file system without the Direct I/O option, use the **vmo** command to adjust the MINPGAHEAD, MAXPGAHEAD, and MAXFREE parameters to achieve the highest sequential I/O throughput.

See also *AIX 5L Version 5.2: Performance Management Guide.*

Other considerations

Performance gains from effective use of the LVM can vary greatly, depending on the LVM you use and the characteristics of the workload. For Decision Support Systems (DSS) workloads, you can see a substantial improvement. For On-line Transaction Processing (OLTP) or mixed workloads, you can still expect significant performance gains.

Using a GPFS file system or RAW LVs compared to JFS

File systems are continually being improved, as are various file system implementations. In some cases, using a file system provides for better I/O performance than RAW devices.

Different vendors implement the file system layer in different ways to exploit the strengths of different disks. This makes it difficult to compare file systems across platforms.

The introduction of LVM interfaces substantially reduces the tasks of configuring and backing up logical disks based on RAW partitions.

Note: The Direct I/O feature, introduced in AIX 5L, allows file systems to have performance characteristics similar to RAW devices.

The degree of performance improvement depends largely on the I/O characteristics of the workload.

If you use a journaled file system (JFS), it is easier to manage and maintain database files than if you use RAW devices. In earlier versions, file systems used to support only buffered read and write operations, so every time data was transferred to or from the I/O subsystem (other than the Oracle buffer cache or SGA), extra AIX file buffer caches were created. This was the main drawback to using a journaled file system, and this is one of the two problems solved by Direct I/O.

Note: Oracle9i Real Application Clusters requires the use of RAW devices or the GPFS file system instead of a JFS or JFS2 journaled file system for database files. Direct I/O is enabled by default when using GPFS.

Using Asynchronous I/O (AIO)

Oracle9i takes full advantage of asynchronous I/O (AIO) provided by AIX, resulting in faster database access. AIO interleaves multiple I/O to improve I/O subsystem throughput. The advantage of AIO is achieved only when data is well distributed among different disks.

Using the LVM and striping enhances the effectiveness of AIO operations. The LVM reduces disk contention by striping data across multiple disk spindles. Using AIO with LVM significantly improves RDBMS performance.

AIX versions 4 and higher support asynchronous I/O (AIO) for database files created both on file system partitions and on RAW devices. AIO on RAW devices is implemented fully into the AIX kernel, and does not require server processes to service the AIO requests.

When using AIO on file systems, the kernel server processes (kproc) control each request from the time a request is taken off the queue until it completes. The kernel server processes are also used in I/O with virtual shared disks (VSDs) with FastPath disabled. By default, FastPath is enabled. The number of kproc servers determines the number of AIO requests that can be executed in the system concurrently, so it is important to tune the number of kproc processes when using file systems to store Oracle9*i* datafiles.

If you are using AIO with VSDs and AIO FastPath enabled (the default), the maximum buddy buffer size must be greater than or equal to 128 KB.

Use one of the following commands to set the number of servers. Once again, this applies only when Asynchronous I/O is used on file systems instead of RAW devices.

```
# smitty aio or
# chdev -l aio0 -a maxservers='m' -a minservers='n'
```

Set the minimum value to the number of servers to be started at system boot. Set the maximum value to the number of servers that can be started in response to a large number of concurrent requests. These parameters apply to files only, they do not apply to RAW devices.

The default value for the minimum number of servers is 1. The default value for the maximum number of servers is 10. These values are usually too low to run the Oracle server on large systems with 4 CPUs or more, if you are not using “kernelized” AIO. Oracle Corporation recommends that you set the values listed in Table 4-5.

Table 4-5 AIO parameters

Parameter	Value
MINSERVERS	Varies depending on the asynchronous requests to the AIO servers on the system. Oracle Corporation recommends an initial value equal to the number of CPUs or 10, whichever is lower. Once an appropriate MAXSERVER value is determined, MINSERVERS can be set to (MAXSERVERS / 2)
MAXSERVERS	The initial value should be set to (10 * number of logical disks). Monitor the actual number of AIO servers started, during a typical workload, using the **pstat** or **ps** commands. If the actual number of active AIO servers is equal to the MAXSERVERS, then the MAXSERVERS value should be increased.
MAXREQS	The initial value should be set to (4 * number of logical disks * queue depth).

The queue depth can be determined by using the command **lsattr -E -l hdiskxx** and is typically 3 for SCSI disks, and 8 or more for SSA and FC attached disks.

If the values of the MAXSERVERS or MAXREQS parameters are set too low, you will see the following warning messages repeated:

```
Warning: lio_listio returned EAGAIN
Performance degradation may be seen
```

You can avoid these errors by increasing the value of the MAXSERVERS parameter to greater than the number of AIO servers running. To display the number of AIO servers running, enter the following command as the root user:

```
# pstat -a | grep -c aios
# ps -k | grep aioserver
```

Check the number of active AIO servers periodically and change the values of the MINSERVERS and MAXSERVERS parameters if necessary. The changes take place when the system restarts.

I/O Slaves

I/O Slaves are specialized Oracle processes that perform only I/O. They are seldom used on AIX, as Asynchronous I/O is the default and recommended way for Oracle to perform I/O operations on AIX.

I/O Slaves are allocated from shared memory buffers. They use a set of initialization parameters, listed in Table 4-6, that allow a degree of control over the way they operate.

Table 4-6 I/O Slaves parameters

Parameter	Range of Values	Default Value
DISK_ASYNCH_IO	TRUE/FALSE	TRUE
TAPE_ASYNCH_IO	TRUE/FALSE	TRUE
BACKUP_TAPE_IO_SLAVES	TRUE/FALSE	FALSE
DBWR_IO_SLAVES	0 - 999	0
DB_WRITER_PROCESSES	1 - 20	1

There may be situations when the use of asynchronous I/O must be turned off. The first two parameters in the preceding table, DISK_ASYNCH_IO and TAPE_ASYNCH_IO, allow asynchronous I/O to be switched off for disk or tape devices, respectively. Because the number of I/O Slaves for each process type defaults to zero, by default no I/O Slaves are deployed.

The DBWR_IO_SLAVES parameter should be set to greater than 0 only if the DISK_ASYNCH_IO or TAPE_ASYNCH_IO parameters have been set to FALSE, otherwise the database writer process (DBWR) becomes a bottleneck. In this case, the optimal value on AIX for the DBWR_IO_SLAVES parameter is 4.

Using the DB_FILE_MULTIBLOCK_READ_COUNT parameter

A large value for the DB_FILE_MULTIBLOCK_READ_COUNT initialization parameter usually yields better I/O throughput on sequential scans. On AIX, this parameter ranges from 1 to 512, but using a value higher than 16 usually does not provide additional performance gains.

Set this parameter so that its value when multiplied by the value of the DB_BLOCK_SIZE parameter produces a number that is larger than the LVM stripe size. Such a setting causes more disks to be used.

Using write behind

The write behind feature enables the operating system to group write I/Os together up to the size of a partition. Doing this increases performance because the number of I/O operations is reduced. The file system divides each file into 16 KB partitions to increase write performance, limit the number of dirty pages in memory, and minimize disk fragmentation. The pages of a particular partition are not written to disk until the program writes the first byte of the next 16 KB partition. To set the size of the buffer for write behind to eight 16 KB partitions, use the following command:

```
# vmo -c 8
```

To disable write behind, enter:

```
# vmo -c 0
```

Tuning sequential read ahead

Note: The following applies only to file systems and when Direct I/O is not used.

The Virtual Memory Manager (VMM) anticipates the need for pages while reading a sequential file. It observes the pattern in which a process accesses a file. When the process accesses two successive pages of the file, the VMM assumes that the program will continue to access the file sequentially, and schedules additional sequential reads of the file. These reads overlap the program processing and make data available to the program sooner. Two

VMM thresholds, implemented as kernel parameters, determine the number of pages it reads ahead:

- MINPGAHEAD - The number of pages read ahead when the VMM first detects the sequential access pattern
- MAXPGAHEAD - The maximum number of pages that VMM reads ahead in a sequential file.

Set the MINPGAHEAD and MAXPGAHEAD parameters to appropriate values for your application. The default values are 2 and 8, respectively. Use the **vmo** (or **vmtune**) command to change these values. You can use higher values for the MAXPGAHEAD parameter in systems where the sequential performance of striped logical volumes is of paramount importance. To set the MINPGAHEAD parameter to 32 pages and the MAXPGAHEAD parameter to 64 pages, enter the following command as the root user:

```
# vmo -r 32 -R 64
```

Set both the MINPGAHEAD and MAXPGAHEAD parameters to a power of two. For example, 2, 4, 8,...512, 1042, and so on.

Tuning disk I/O pacing

Disk I/O pacing is an AIX mechanism that allows the system administrator to limit the number of pending I/O requests to a file. This prevents disk I/O-intensive processes from saturating the CPU. Therefore, the response time of interactive and CPU-intensive processes does not deteriorate.

You can configure disk I/O pacing by adjusting two system (sys0) parameters: the high-water mark and the low-water mark. When a process writes to a file that already has a pending high-water mark I/O request, the process is put to sleep. The process wakes up when the number of outstanding I/O requests falls below or equals the low-water mark.

You can use the SMIT tool to change the high- and low-water marks. Determine the water marks through trial-and-error. Use caution when setting the water marks because they affect performance. Tuning the high- and low-water marks has less of an effect on disk I/O operations with blocks larger than 4 KB.

4.4.3 Resilvering with Oracle9*i*

If you disable mirror write consistency (MWC) for an Oracle data file allocated on a RAW logical volume (LV), the Oracle9*i* crash recovery process uses resilvering to recover after a system crash. This resilvering process prevents database inconsistencies or corruption.

During crash recovery, if a data file is allocated on a logical volume with more than one copy, the resilvering process performs a checksum on the data blocks of all of the copies. It then performs one of the following:

- If the data blocks in a copy have valid checksums, the resilvering process uses that copy to update the copies that have invalid checksums.
- If all copies have blocks with invalid checksums, the resilvering process rebuilds the blocks using information from the redo log file. It then writes the datafile to the logical volume and updates all of the copies.

On AIX, the resilvering process works only for datafiles allocated on RAW logical volumes for which MWC is disabled. Resilvering is not required for data files on mirrored logical volumes with MWC enabled, because MWC ensures that all copies are synchronized.

If the system crashes while you are upgrading a previous release of Oracle9*i* that used data files on logical volumes for which MWC was disabled, use the **syncvg** command to synchronize the mirrored LV before starting the Oracle server. If you do not synchronize the mirrored LV before starting the server, Oracle might read incorrect data from an LV copy.

Note: If a disk drive fails, resilvering does not occur. You must enter the **syncvg** command before you can reactivate the LV.

Caution: Oracle Corporation supports resilvering for data files only. Do not disable MWC for redo log files.

4.4.4 CPU scheduling and process priorities

The CPU is another system component for which processes might contend. Although the AIX kernel allocates CPU effectively most of the time, many processes compete for CPU cycles. If your system has more than one CPU (SMP), there might be different levels of contention on each CPU.

Changing process running time slice

The default value for the runtime slice of the AIX RR dispatcher is ten milliseconds. Use the **schedo** command to change the time slice. However, be careful when using this command. A longer time slice causes a lower context switch rate if the applications' average voluntary switch rate is lower. As a result, fewer CPU cycles are spent on context-switching for a process and the system throughput should improve.

However, a longer runtime slice can deteriorate response time, especially on a uniprocessor system. The default runtime slice is usually acceptable for most applications. When the run queue is high and most of the applications and Oracle shadow processes are capable of running a much longer duration, you might want to increase the time slice by entering the following command on AIX 5.2 and later:

```
# /usr/sbin/schedo -o timeslice=n
```

In the previous example, choosing a value of 0 for `n` results in a slice of 10 milliseconds (ms), choosing a value of 1 results in a slice of 20 ms, choosing a value of 2 results in a slice of 30 ms, and so on.

Using processor binding on SMP systems

Binding certain processes to a processor can improve performance substantially on an SMP system. Processor binding is available and fully functional with AIX Version 4 and higher.

Processor binding offers the following benefits:

- Provides higher-priority applications with a relatively larger share of CPU time.
- Maintains the process context for a longer period.

Processor binding has the following drawbacks:

- A CPU cannot be dynamically removed from a system if it has processes bound to it.
- Processor binding on AIX is not automatic. On a multiprocessor system, you must explicitly bind a process to a processor by using the bindprocessor command. Only the root user or the Oracle software owner can bind an Oracle process to a processor. The child processes inherit the processor binding.

Oracle Corporation recommends binding the various Oracle background processes (except the database writer process) to different processors and leaving one processor free to service the database writer process. This guarantees the database writer a processor on which to execute and at the same time allows the database writer process to migrate freely to the other processors if it becomes CPU-bound.

Note: Processor binding is a complicated issue and it should be handled with care. Processes bound to a processor cannot migrate to different processors even if these processors are free. This might degrade application performance. An environment of homogenous applications with a balanced load is more suitable for processor binding.

The binding of a process to a processor is not exclusive. The processor is free to execute other processes.

Processor binding in a networked client and server environment

When an Oracle client process connects to an Oracle server process using an Oracle Net Services listener, the server process can easily be bound to a processor by binding the listener process. All Oracle server processes that the listener subsequently spawns are bound to the same processor.

One way to do this is to start multiple listeners, each listening on its own port. You must customize the $ORACLE_HOME/network/admin/listener.ora file to have one set of lines for each listener. Launch multiple listeners on the server side.

On the client side, you might want to customize the tnsnames.ora file so that clients or even applications connect to different ports that are listened on by different listeners. For example, you can modify the listener.ora file and have two listeners, L1 and L2, that listen on ports 1521 and 1522, respectively, as follows:

1. Modify the listener.ora file as shown in Example 4-20.

Example 4-20 Binding a listener process

```
L1 =
  (ADDRESS_LIST =
     (ADDRESS= (PROTOCOL= TCP)(Host= nowhere)(Port= 1521))
  )
SID_LIST_L1 =
  (SID_LIST =
    (SID_DESC =
      (ORACLE_HOME= /oracle)
      (SID_NAME = ordb)
    )
  )
L2 =
  (ADDRESS_LIST =
      (ADDRESS= (PROTOCOL= TCP)(Host= nowhere)(Port= 1522))
  )
SID_LIST_L2 =
  (SID_LIST =
    (SID_DESC =
      (ORACLE_HOME= /oracle)
      (SID_NAME = ordb)
    )
  )
```

2. Start the following two listeners:

```
$ lsnrctl start L1
$ lsnrctl start L2
```

3. Determine the process IDs for the two listeners:

```
$ ps -ef | grep tnslsnr
```

4. Bind the listener processes to particular processors:

```
$ bindprocessor process_id_for_L1 id_for_process1
$ bindprocessor process_id_for_L2 id_for_process2
```

In the preceding example, id_for_process1 is 0, id_for_process2 is 1, and so on.

Using this method, all Oracle server processes that handle communications with a particular client run on the same processor.

Processor binding in a local environment

Processor binding is more difficult when the clients and the Oracle servers run on the same computer using the two-task pipe driver. You must determine the process ID for each server process and manually bind it to a processor. The administrative overhead is excessive and probably not worth the effort unless the servers have long process lives.

Processor binding of Oracle processes can have negative effects on the performance of other applications running on the same system. Careful tuning and monitoring is strongly recommended.

4.4.5 Oracle9i Real Application Clusters and HACMP/ES

Network tuning for Transparent Application Failover

In cases where you are experiencing a Transparent Application Failover time of over 10 minutes, consider tuning network parameters rto_length, rto_low, and rto_high to reduce the failover time.

The issue has to do with a TCP time-out and retransmission problem in which clients who are currently connected to a crashed node are not getting acknowledgement from the failed instance. Consequently, the client continues to retransmit the same packet over and over again using an algorithm called Exponential Backoff (refer to any TCP/IP book for more details).

On AIX, the default time-out value is set to approximately 9 minutes. However, this parameter is tunable via “no” using the load time attributes—rto_length, rto_low, and rto_high. Using these parameters, you can control “how often” and “how many times” a client should retransmit the same packet before it gives up. The parameter rto_low (default is 1 sec) and rto_high (default is 64 sec) controls “how often to transmit the packet”, while rto_length (default is 13) controls “how many times to transmit the packet”.

For example:

Using the Exponential Backoff algorithm, if we leave the AIX default as above, the time-out value is set to approximately 9.3 minutes. However, using the same algorithm, and setting rto_length to 7, the timeout value goes down to 2.5 minutes.

Note: You should check the quality of your network transmissions before setting any of the above parameters. This can be done using the **netstat** command. If the quality of network transmissions is bad, you might require a longer time-out value.

HACMP and PSSP on the same node

When HACMP and PSSP are installed on the same machine, Oracle9*i* assumes that customers want to use PSSP functionality. In the default configuration, customers using Oracle9i Real Application Clusters must place all database files on Virtual Shared Disks (VSDs). If you wish to use the Concurrent Logical Volume Manager (CLVM) instead of VSDs, then enable HACMP functionality by setting the environment variable PGSD_SUBSYS to grpsvcs. Oracle9*i* will not allow VSDs and Concurrent Logical Volumes (CLVs) to be used on the same database. If PSSP services are being used, Oracle9*i* will report an error if the customer attempts to use CLVs. If HACMP services are used (that is, PGSD_SUBSYS is set to grpsvcs), Oracle9*i* will report an error if the customer attempts to use VSDs.

The PGSD_SUBSYS environment variable should be set in all the environments where Oracle9*i* is used, including the listener.ora file and the information repository for the database server configuration.

To enable HACMP functionality on a node that has both PSSP and HACMP installed, the information repository for the database server configuration should be updated to include the environment PGSD_SUBSYS=grpsvcs:

```
$ svrctl set env -p db_name -t PGSD_SUBSYS=grpsvcs
$ srvconfig -conv $ORACLE_HOME/ops/db_name.conf
```

where `db_name.conf` is the name of your Oracle8*i* OPSCONF configuration file (its equivalent in Oracle9*i* RAC is the destination file specified in /var/opt/oracle/srvConfig.loc).

Note: If you are upgrading from an Oracle8*i* Parallel Server database, you can use the **`svrconfig -conv`** command to convert your database configuration file, db_name.conf, to a shared RAW device for the GSD in your Oracle9*i* Real Application Clusters database.

4.4.6 Oracle9i backup issues

Oracle Corporation recommends that you use Recovery Manager (RMAN) to back up RAW devices. If you do use the **`dd`** command to back up RAW devices, use it with caution, as follows:

The offset of the first Oracle block on a RAW device may be 0, 4K or 128K, depending on the device type. The command'**`offset`** can be used to determine the proper offset.

To back up the RAW device to tape, enter a command similar to the following:

```
$ dd if=/dev/raw_device of=/dev/rmt0.1 bs=256k
```

To restore the RAW device from tape, enter commands similar to the following:

```
$ dd if=/dev/rmt0.1 of=/dev/raw_device count=63 seek=1 skip=1 bs=4k
$ mt -f /dev/rmt0.1 bsf 1
$ dd if=/dev/rmt0.1 of=/dev/raw_device seek=1 skip=1 bs=256k
```

Note: For HSDs, do not skip the first 4 KB. Use the following command to restore the RAW device, instead of the three preceding commands:

```
$ dd if=/dev/rmt0.1 of=/dev/raw_device bs=256K
```

5

Implementing RAC over GPFS

This chapter provides the background, guidelines, recommendations and samples of how to implement an Oracle9*i* RAC (Real Application Clusters) database, using General Parallel File System (GPFS).

The following topics are discussed:

- Benefits of RAC implementation with GPFS
- Oracle9*i* RAC overview
- Cluster planning
- Physical database design
- RAC implementation steps
- RAC client side failover and load balancing configuration
- Troubleshooting
- Performance tuning
- References

5.1 Benefits of Oracle9*i* RAC implementation with GPFS

The main benefits of an Oracle9*i* RAC implementation over GPFS are:

- It enables a file system database implementation

  In the past, only RAW devices (logical volumes) were supported under HACMP/ESCRM for the implementation of Oracle9*i* RAC or Oracle Parallel Server (OPS).

  For many customers, a file system database implementation is preferred because this simplifies system administrators' and DBA's tasks to pre-allocate RAW devices for the physical database implementation (using the AUTOEXTEND attribute for the tablespaces), and system administration tasks (export, log archiving, backup, etc.).

  Unlike most UNIX file systems, which are designed for a single server environment, GPFS allows parallel applications to simultaneously access the same files from any node in the GPFS node set. The shared access GPFS is capable of holding the database files, control files, and redo log files required by Oracle9*i* RAC. It satisfies the Oracle9*i* RAC shared disk requirement.

- It provides RAC with striping I/O performance

  GPFS is designed to provide high performance by striping I/O across the storage subsystem. GPFS stripes the data across all disks that are part of a file system.

- It provides Direct I/O with close to RAW device performance

  When RAW devices are used, the Oracle bypasses the file buffer cache and accesses the logical volume directly. The RAW logical volume I/O eliminates the file system overhead, and the need for AIX virtual memory tuning.

  This enables Oracle to allocate a larger buffer cache for caching more data in a more direct way. For I/O intensive applications, the use of RAW devices for Oracle datafiles generally results in performance improvement.

  For RAC implementation using GPFS, Oracle opens GPFS with Direct I/O that bypasses the file cache and transfers data directly from disk into the user space buffer.

  Although performance estimates for running RAC on GPFS can be very close to the performance of Oracle using RAW devices, some degradation may be expected, since GPFS introduces another software layer in the I/O path.

- It increases data availability and failure survivability

  GPFS also provides high availability through logging and replication. From the cluster perspective, if the cluster loses a node, the GPFS file system availability on the rest of the nodes will not be affected as long as the GPFS quorum is met.

5.2 Oracle9*i* RAC overview

The Oracle9*i* RAC database is a shared disk cluster architecture. It allows multiple Oracle instances to execute against the same database. An Oracle instance consists of a System Global Area (SGA) and the Oracle background processes.

The typical installation involves a cluster of nodes with access to a set of shared disks. A node is defined as a collection of processors, shared random access memory (RAM), and disks that store an instance's data and code.

The shared disks store the system tablespace, online redo logs, undo tablespaces, control files, and the database files. The Oracle executable can be saved either locally on each node's local file system, or also on the shared disks.

Real Application Clusters provide the following advantages over the single instance databases:

- High availability - clients and applications can access any database instance.
- Scalability - allows nodes (thus instances) to be added for scalability.
- Transparency - applications run on a “single image system.”

To implement a RAC using GPFS, you need to have some basic understanding of the Oracle9*i* RAC components. The following high-level Oracle9*i* overview is written with the intent to assist further discussions later in the chapter.

Oracle processes

The processes in an Oracle system can be categorized into two major groups:

- User processes that run the application or Oracle tool code.
- Oracle processes that run the Oracle server code. These include:
  - Server processes
  - Background processes

For details, see Figure 5-1 on page 148 where:

- PMON - Process Monitor
- SMON - System Monitor
- DBRW - Database Writer
- LOGW - Log Writer
- LMON - Lock Monitor Process (Global Enqueue Service Monitor).
- GES - Global Enqueue Service Daemon.
- LMS - Lock Manager Server (Global Cache Service Processes).
- GCS - Global Cache Service.
- LMD - Lock Manager Daemon (Global Enqueue Service Daemon)
- LCK*n* - Lock Process
- NPIC - Network Inter-Process Communication
- CM - Cluster Manager
- NM - Node Monitor
- GSD - Global Services Daemon

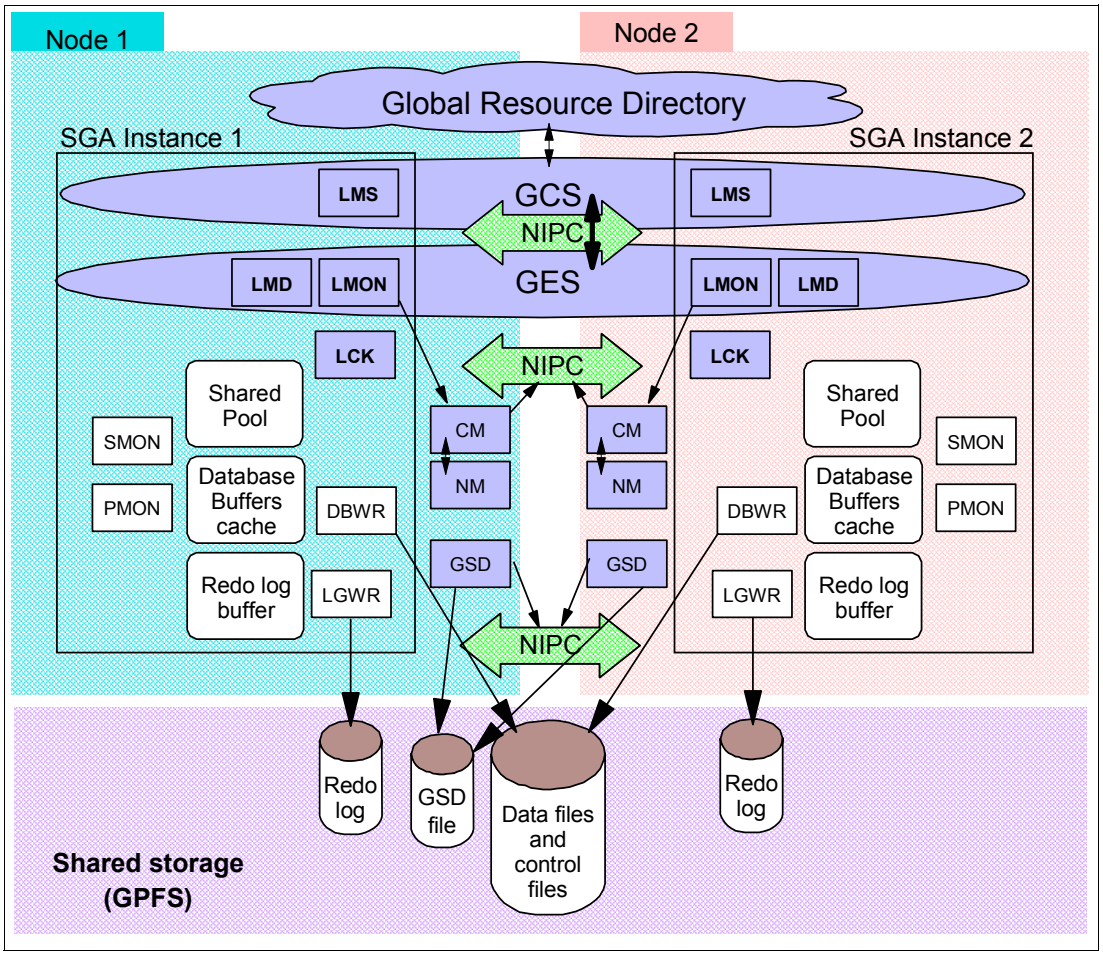


Figure 5-1 Oracle9i RAC functional diagram

5.2.1 Oracle9*i* RAC Cache Fusion

In Oracle 8*i*, the consistent read cache fusion was introduced in Oracle Parallel Server (OPS). It allowed a consistent block image to be transferred from the buffer cache of the writing instance to the cache of the reading instance using the cluster interconnect network. This is known as Cache Fusion, Phase 1, or Write-Read Cache Fusion.

Oracle9*i* RAC supports full Cache Fusion, which utilizes all the collection caches made available by all nodes in the cluster, to satisfy the database requests.

With Cache Fusion, modified blocks currently in the buffer cache of one instance are transferred to another instance across the interconnect rather than forcing disk writes of the blocks.

Requests for a data block are satisfied by a local cache, then by a remote cache before the disk read. Disk I/O operations are only performed when the data blocks are not available in the collective cache. This is valid for:

- Write-write transfer

  An instance requests a database block for write when this block has previously been updated by another instance.

- Write-read transfer

  An instance requests a database block for write when the block has previously been read by another instance.

- Read-read transfer

  An instance requests a database block for read when this block has previously been read by another instance.

- Read-write transfer

  An instance requests a database block for read when this block has previously been updated by another instance.

Prior to Oracle9*i* RAC, the Oracle Parallel Server (OPS) cluster was considered most suitable for data warehouse (DWH) deployment (which is mainly query oriented).

Oracle9*i* RAC, since it supports full Cache Fusion, and with its built-in node failure high availability feature, is now also suitable for a true OLTP environment.

5.2.2 Cluster interconnect network considerations

To ensure that the Cache Fusion provides the highest level of Oracle block transfer rate, you need to understand how RAC uses these interconnect networks, and how to name these RAC interconnects in an HACMP/ES cluster environment.

These are the Oracle9*i* RAC interconnect network requirements and considerations:

- Fast interconnect with low latency.

  Oracle9i RAC Cache Fusion allows any Oracle block being held by an instance to be copied to another instance across the cluster interconnect. It allows data to be shared between instances without the overhead of forced disk block write transfer. To ensure the highest level of transfer, the cluster interconnect should be based on a fast interconnect network, with low latency.

- Redundant RAC interconnects are recommended.

  To eliminate the network single point of failure, the RAC cluster interconnect should have redundant network connections.

- A dedicated RAC interconnect network is highly recommended.

  The RAC interconnect networks should be dedicated for Oracle usage. For high performance, one should never be mixed with the client networks.

Oracle Transparent Network Failover Failback (TNFF)

Oracle offers a high availability feature called Oracle Transparent Network Failover Failback (TNFF). In an IBM pSeries environment, this works closely with HACMP/ES.

Tip: The Oracle9*i* RAC network selection mechanism must be well understood before planning and configuring HACMP/ES.

Interconnect network selection

The Oracle cluster interconnect networks can be configured using either a “private” or a “public” network type defined in HCMP/ES. The Oracle IPC traffic is sent over one of the networks. Oracle RAC chooses the network interface according to the following rules:

1. HACMP “service” networks are required.

   Oracle chooses only HACMP/ES networks configured as “service”.

2. Oracle uses the HACMP cllsif utility to determine the network interface.

   Oracle uses the HACMP utility cllsif to determine which network interface to use based on the rules described below. Oracle chooses the network to use according to the cllsif alphanumeric sort list.

3. Private networks are preferred over public networks.

   If there are multiple private networks under HACMP, RAC chooses the interconnect communication by performance quality in the following order: SPswitch > FDDI > Ethernet

   Note: Although networks can be labeled inside HACMP as public or private, this attribute has no functional meaning. It's only a way for Oracle to make network selections.

4. Maximum is three networks.

   Oracle will choose up to three networks. Oracle recommends two private and one (or more) public networks for the cluster interconnect.

The Network failover and failback sequence

A typical configuration, using common network hardware, has two private interconnects and one public interconnect scenarios.

If the first HACMP/ES private service network goes down, Oracle TNFF fails over to the second private service network. If the second private service network also fails, TNFF continues to fail over to the public network.

If any of the two failed private interconnects is restored, the TNFF fails back to that private service interconnect. If both failed private interconnects are fixed, then TNFF restores the RAC communication to the first listed private service network, according to the **`cllsif`** command output alphanumeric sort order.

A serial link or a non-IP network is recommended

A non-IP network, such as RS232, target mode SSA, is also highly recommended for the HACMP cluster. Although Oracle does not use this network, it helps to keep the HACMP cluster up if all IP networks are down.

The parameter CLUSTER_INTERCONNECT in init.ora disables the TNFF

Using the CLUSTER_INTERCONNECTS setting in the Oracle initialization parameter file, you can explicitly specify the interface to be used. However, by specifying this, the Oracle Transparent Network Failover Failback (TNFF) is disabled.

In this case, an interconnect failure that is normally unnoticeable, would instead cause an Oracle cluster failure. This is because Oracle still attempts to access the network interface, which has gone down.

5.2.3 Dynamic System Global Area (SGA)

The System Global Area (SGA) is a shared memory that contains data and control information for an Oracle instance. Oracle allocates the SGA when the instance starts and deallocates it when the instance shuts down. Each of the RAC instances has its own SGA.

The size of the Oracle9*i* SGA is determined by:

- DB_CACHE_SIZE

- LOG_BUFFER
- SHARED_POOL_SIZE
- LARGE_POOL_SIZE

Prior to Oracle9*i,* the SGA was always a static allocation of memory. The size of memory allocated for SGA is calculated based on values in the init.ora parameter file.

Once allocated, the amount of used memory cannot be dynamically adjusted. If a DBA wants to change any of the SGA components, such as buffer cache size, he needs to first shut down the Oracle instance, modify the corresponding init.ora parameter, and then restart the Oracle instance for the new size to take effect.

The Dynamic SGA, a new Oracle9*i* feature, allows Oracle to set, at runtime, the amount of virtual memory Oracle uses for the SGA. Oracle can start the instance underconfigured (with a smaller amount of memory) and allow the instance to use more memory by growing the SGA components, up to a maximum of SGA_MAX_SIZE. This allows the SGA to be dynamically changed without shutting down the instance. If SGA_MAX_SIZE, specified in the initialization parameter file, is less than the sum of all components specified or defaulted at initialization time, then the setting in the initialization parameter file is ignored.

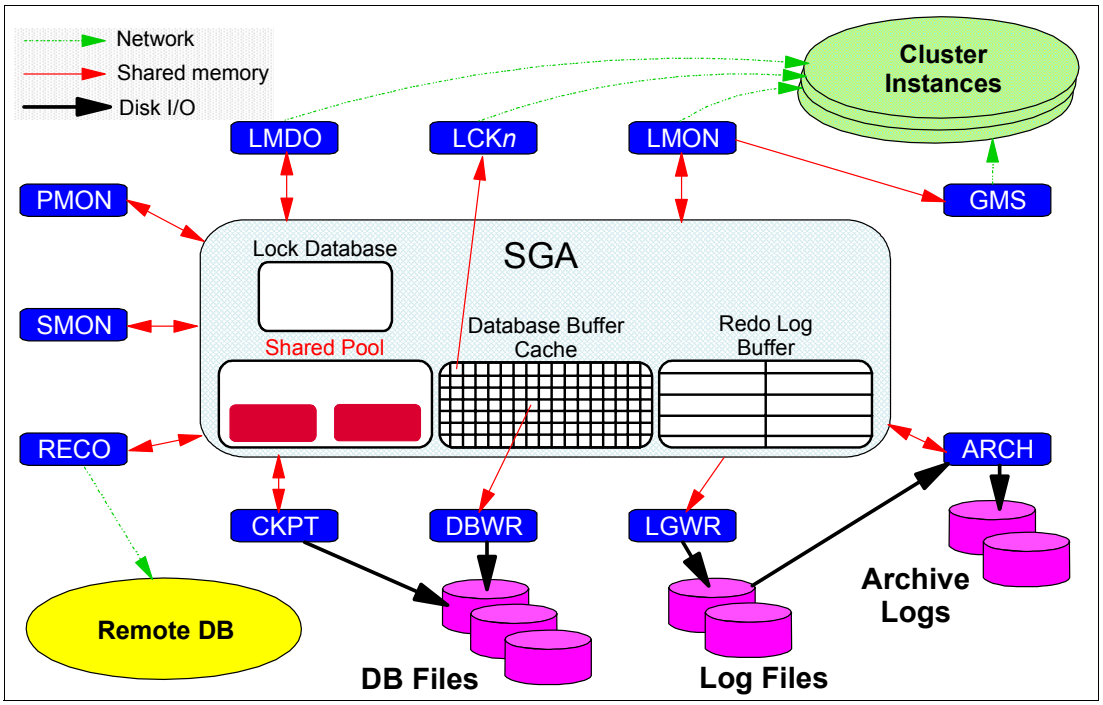


Figure 5-2 Oracle9i SGA and background processes

5.2.4 Program Global Area (PGA) aggregate target

A Program Global Area (PGA) is a memory region that contains data and control information for a server process. It is a non-shared memory created by Oracle when a server process is started. Access to it is exclusive to that server process. The total PGA memory allocated by each server process attached to an Oracle instance is also referred to as the aggregated PGA memory allocated by the instance.

To simplify and improve memory usage, Oracle9*i* introduces a new system level initialization parameter, called PGA_AGGREGATE_TARGET. It specifies the “target” aggregate amount of

PGA memory available to the instance. This parameter is only a target and can be dynamically modified at the instance level by DBA.

To enable PGA_AGGREGATE_TARGET, the initialization parameter WORKAREA_SIZE_POLICY has to be set to AUTO. This allows Oracle to determine each work area automatically.

5.2.5 Undo management

The conventional rollback segment *undo* requires a considerable amount of effort to manage and monitor. For example, the rollback segments have to be created, sized properly and continuously tuned. DBA also needs to deal with issues including *out of space* and *snapshot too old* problems. Successfully managing rollback segments requires in-depth understanding of the application logic and how undo management works.

The Oracle9*i* new feature Automatic Undo Management simplifies the process of managing undo segments by internalizing the performance of these segments. The UNDO_MANAGEMENT = AUTO initialization parameter needs to be set to cause the instance startup command to activate an instance in the Automatic Management mode. The default value of this parameter is “manual”.

When the instance starts up, it uses the undo tablespace specified by the UNDO_TABLESPACE parameter. With automatic undo management, you cannot create, drop, or alter undo segments, which may not be beneficial in a cluster database environment.

Note: For Oracle9*i* RAC, each instance should have its own UNDO_TABLESPACE. This is normally being set in the instance-specific init.ora.

5.2.6 Redo log threads

Redo logs contain records of all changes made to data in the database buffer cache. If an instance failure occurs, the redo log files are used to recover the modified data that was in memory.

In Real Application Clusters, each instance writes to its own set of online redo log files. The redo information written by a single instance is called a “thread of redo,” which consists of its own online redo log groups. Each online redo log file is associated with a particular thread number.

5.2.7 Oracle DB_BLOCK_SIZE

The parameter DB_BLOCK_SIZE specifies the size (in bytes) of Oracle database blocks. The value for DB_BLOCK_SIZE in effect at the time you create the database determines the size of the blocks.

Supported DB block size values are 2 KB, 4 KB, 8 KB, 16 KB, and 32 KB. For a RAC implemented using GPFS, we recommend 16 KB for DB_BLOCK_SIZE, the same as the GPFS data block size. To avoid GPFS partial block write, any DB_BLOCK_SIZE less than 16 KB is not recommended. See also “GPFS block size” on page 161.

5.2.8 Tablespace

The tablespace is a database storage unit that groups related logical structures together. The Oracle database is logically divided into one or more tablespaces. One or more datafiles are

explicitly created for each tablespace to physically store the data. Some of the basic tablespaces are as follows:

- System tablespace

  The data dictionary is stored in the SYSTEM tablespace. The source code for packages, procedures, functions, triggers, and methods resides in the data dictionary tables. If your application makes extensive use of these objects, you may need to increase the size of the SYSTEM tablespace.

- Temporary tablespace

  Temporary tablespaces can be “dictionary managed” or “locally managed”. Starting with Oracle9*i*, Oracle recommends the use of locally managed temporary tablespaces. These have temporary datafiles, tempfiles, which are set to *nologging* (operation on objects contained in these temporary files is not logged, as with the normal database files). To create temporary tablespaces, use the database client and the CREATE TEMPORARY TABLESPACE statement.

- Undo tablespace

  Undo tablespaces are special tablespaces used solely for storing undo information (undo information is used to recover from transaction failures). Refer to 5.2.5, “Undo management” on page 152.

- Automatic Segment Space Management (ASSM) tablespace

  Automatic Segment Space Management (ASSM) is a new Oracle9*i* scheme of managing free space inside the database segments. It provides the capability to track in-segment free and used space through *bitmaps*, as opposed to the previously used *freelists*. ASSM provides better space utilization, better concurrency handling.

  The purpose of ASSM is to provide a simpler way of managing segment space, resulting in better space utilization, and eliminating any need to specify PCTUSED, FREELISTS, and FREELISTS GROUPS attributes for segments.

  ASSM improves the performance of concurrent Data Manipulation Language (DML) operations due to the fact that different parts of the bitmap can be used simultaneously, thus eliminating serialization for free space lookup.

  The performance and manageability provided by ASSM may be particularly noticeable in the RAC environment. It removes the need to alter the number of FREELISTS and FREELIST GROUPS when a new instance is brought online, which saves the downtime associated with table reorganizations.

5.2.9 Control files

Control files record the physical structure of a database and contain the database name, the names and locations of associated databases, the online redo log files, the timestamp of the database creation, the current log sequence numbers, and checkpoint information.

Each database should have at least two control files.

5.2.10 Initialization parameters

Initialization parameter file (static server)

An initialization parameter file is a text file that contains a list of initialization parameters and Oracle database instance uses when it is started (created). The name of the initialization parameter file may vary. The typical initialization parameter file name is init<SID>.ora, which identifies the initialization parameters for a particular instance. SID is the instance identifier, and can be set up in the ORACLE_SID variable.

This parameter file can be placed under any directory, but at instance startup time, you need to use the pfile=<path/file> parameter to specify where this file is located. If `pfile=` is not specified, then its default location should be $ORACLE_HOME/dbs.

Dynamic server initialization parameter file (SPFILE)

Oracle9*i* introduces a new parameter file named SPFILE to store the instance parameters. Using this file, it provides the ability to dynamically change some of the initialization parameters. The SPFILE is a binary file that *must not* be edited manually, using a text editor. Manually editing this file will result in a corrupted file.

This file is maintained by the Oracle server. By default, this file is located in $ORACLE_HOME/dbs. The SPFILE is created from an init<SID>.ora using the **`create spfile`** Oracle client command. It can be used to designate both global and instance-specific settings.

Oracle startup command searching sequence

If you start an instance without specifying the pfile parameter, Oracle will search the default location in the following order (SQL> startup):

- spfile<$ORACLE_SID>.ora
- spfile.ora
- init<$ORACLE_SID>.ora

5.3 Environment planning

5.3.1 pSeries hardware planning

pSeries 64-bit hardware

Oracle9*i* requires 64-bit pSeries hardware, which can be verified by executing one of the following commands:

- `bootinfo -y`
- `getconf HARDWARE_BITMODE`

  (This command is available only in AIX 5.2.)

pSeries server I/O requirement

- Gather the I/O requirement, such as bandwidth and throughput, for both network traffic and storage I/O.
- Select the pSeries severs to match the I/O requirements.

  Each pSeries server model has its own required and optional number of I/O drawers, and maximum PCI or PCI-X slots. From this information, you can estimate the server I/O bandwidth. Select the pSeries sever that matches your I/O requirements.

  Since RAC requires at least two pSeries servers, if the I/O server bandwidth of the two combined servers does not satisfy the I/O requirement, then you need more servers for the RAC cluster. We recommend large SMP machines for the cluster nodes in a RAC environment, using as few nodes as possible.
- Determine the number of I/O drawers required for each pSeries server.
- Determine the number of disk adapters required for each I/O drawer. If multiple drawers are involved, make sure the I/O adapters are evenly spread out between all I/O drawers.

- Adapter bandwidth performance information is only an estimate. Actual performance is dependant on many factors, such as workload, protocol, tuning, contention with other devices. Safety factor should be embedded in your estimation. Follow these general rules of thumb:

  Note: Always refer to the hardware architecture of your servers, to match the adapters with the I/O capability of the PCI bus inside the servers.

  - Two 1-Gigabit Fibre Channel (data intensive): 150 MB/s
  - One 1-Gigabit Fibre Channel adapter (data intensive): 85 MB/s
  - Advanced Serial RAID plus adapter: 90 MB/sec
- Select the storage subsystem

  Be sure to understand your storage architecture.

pSeries networks

The most commonly used Oracle9*i* networks on pSeries are:

- SP Switch2
- SP Switch
- Gigabit Ethernet
- 100 Mb Ethernet

EtherChannel

Besides the previously listed network types, EtherChannel can also be considered.

EtherChannel is a network port aggregation technology that allows several Ethernet adapters to be put together to form a single pseudo Ethernet device.

For example, ent1, ent2, ent4, and ent5 AIX devices can be aggregated to form the logical device called ent6; the interface ent6 would then be configured with an IP address, as shown in Figure 5-3 on page 156. The system considers these aggregated adapters as one logical device (adapter).

Therefore, IP is configured over the logical adapter as with any other Ethernet adapter. In addition, all adapters in the EtherChannel use the same hardware (MAC) address, so they are treated by remote systems as a single adapter.

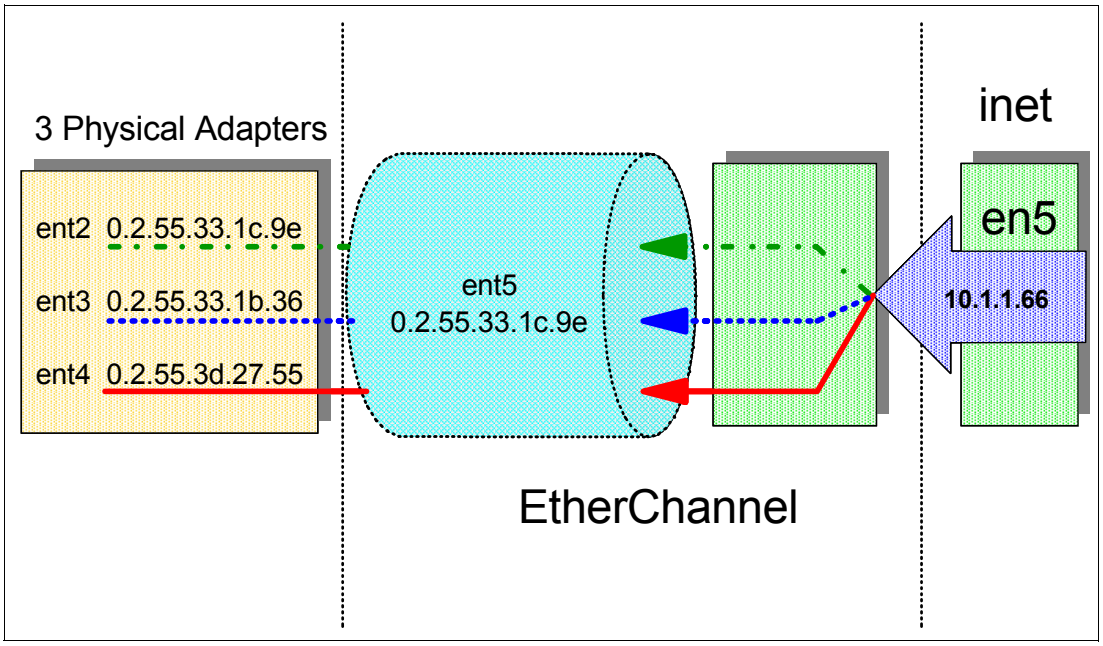


Figure 5-3 EtherChannel aggregation

The major benefit of the EtherChannel is that it provides the network bandwidth of the combined adapters, but in a single network entity (bandwidth aggregation). Also, if one of the adapters fails, the packets are automatically sent over the remaining adapters, without disruption to existing user connections (availability). The adapter is automatically returned to service on the EtherChannel or Link Aggregation when it is recovered.

EtherChannel software support was first shipped in AIX 4.3.3. Its primary function is to provide bandwidth aggregations for the network. It supports Fast Ethernet (100baseT), IBM Gigabit Ethernet-SX PCI adapter, IBM 4-port 10/100 Base-TX Ethernet PCI adapter, and IBM 10/100/1000 base-T Ethernet PCI adapter. EtherChannel is a software implementation at the AIX level, and currently supports up to eight adapters.

There are three supported configuration modes for EtherChannel:

- Standard
- Round Robin
- IEEE 802.3ad

In our test environment we configured an Oracle9*i* RAC private interconnect network over an EtherChannel using the Round Robin algorithm.

Restriction: Even though the EtherChannel acts as a single network interface for the application, the clustering infrastructure is not able to detect any failure of the physical adapters, unless all adapters in an EtherChannel fail, bringing the logical device (adapter) down. The only way to get this information is from the AIX error reporting subsystem.

5.3.2 AIX planning

AIX version

Oracle9*i* Release 2 runs under AIX 4.3.3, AIX 5.1, and AIX 5.2.

AIX kernel

Both AIX 5L 32-bit and 64-bit kernels are supported for Oracle9*i*. Unlike the other UNIXs, the 32-bit AIX kernel can simultaneously and natively execute both 32-bit and 64-bit user-mode applications. The 64-bit kernel is only required when the system has more than 96 GB of physical memory (and, of course, for performance reasons).

To check the AIX kernel type, use the following commands:

```
bootinfo -K
getconf KERNEL_BITMODE
```

(This command is in AIX 5.2 only.)

To enable 64-bit application mode, you can use the fast path **`smitty load64bit`**. Note that the bos.64bit must be installed before the 64-bit application mode can be enabled.

Oracle9*i* ships with both 32- and 64-bit client code. Although the database is 64-bit software, it interoperates with both the 32- and 64-bit client applications.

AIX maximum number of processes allowed per user

The AIX default is 128. This setting needs to be adjusted higher especially for the OLTP environment. Change it to 1024 or higher, according to your application needs. To change this value, you can use one of the following:

```
chdev -l sys0 -a maxuproc='1024'
smitty -> system environment -> 'change / show'
```

5.3.3 Highly available RAC planning

In an AIX environment, Oracle9*i* RAC uses the clustering infrastructure provided by IBM, such as PSSP or HACMP. See Figure 5-4 for an HACMP overview.

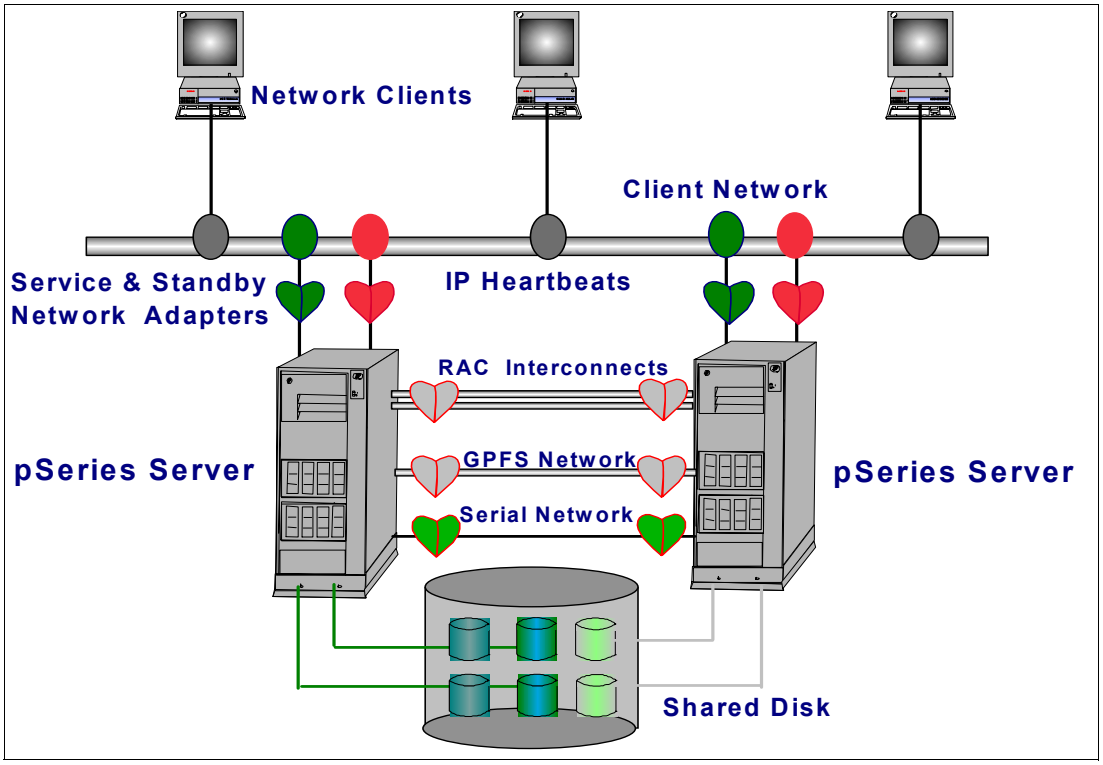


Figure 5-4 Oracle9i RAC high availability cluster with HACMP/ES

HACMP/ES cluster is a prerequisite for RAC in a non-PSSP environment

A basic HACMP/ES is required for RAC because RAC needs to interface with the Event Management layer provided in the HACMP package. To configure a basic HACMP/ES cluster, do the following:

- Configure the HACMP cluster.
- Add cluster nodes to the cluster.
- Configure RAC private interconnects and public networks to the cluster.
- Configure all network adapters to the cluster.
- Synchronize the cluster topology to all cluster nodes.
- Bring up the HACMP cluster services (no resource groups, no IP address takeover (IPAT) needed). This is required for Oracle9*i* RAC operation, including installation.

RAC provides node failure capability

A shared disk is required for RAC to provide protection in case of a node failure. Since every instance has access to all the datafiles in the database, if any of the Oracle instances on a physical node fails, the rest of the surviving Oracle instances will perform recovery.

Highly available RAC interconnect design considerations

Important: Since RAC interconnects are dedicated networks, and Oracle has the Transparent Network Failover Failback (TNFF) feature, we recommend that HACMP IPAT or IP Aliasing are *not* used for RAC interconnect failover.

HACMP/ES should monitor the RAC private interconnects.

It is mandatory that the RAC interconnect failover be handled only by the Oracle Transparent Network Failover Failback (TNFF) mechanism.

We recommend either of the following two RAC interconnect options:

Option 1: Two private interconnects with one primary and one standby

For this option, two private HACMP networks are used for the RAC interconnects. Both private interconnects should be defined under HACMP/ES with the “service” attribute. Oracle uses the HACMP/ES utility **`cllsif`** to determine the networks to be used for RAC interconnect. Since **`cllsif`** does an alphanumeric sort of the network adapter names, Oracle will choose the first listed “service” adapter on the first “private” network for RAC interconnect.

If the first private service interconnect fails, then Oracle Transparent Network Failover Failback (TNFF) fails over to the second private service interconnect. If both private interconnects fail, Oracle TNFF fails over to the public client network (the same alphanumeric sort order is used for “picking” the public network).

Option 2: EtherChannel with round robin over multiple interconnects

You can also use EtherChannel to bundle all the intended cluster interconnect adapters into one pseudo interface. The number of the interconnect adapters required depends on the cache fusion requirements in your environment. The AIX EtherChannel supports up to eight aggregate adapters per link.

We tested the EtherChannel round robin algorithm. It can be used to evenly distribute the load over the links within the EtherChannel, which aggregates the bandwidth of all links. The EtherChannel hash algorithm may skew the load to a certain link in a cluster if there are only a few nodes.

If EtherChannel is used, since it has high availability built in, any of the EtherChannel link failures will be transparently handled by AIX. As long as one physical link is up, EtherChannel continues to function. If the entire EtherChannel fails, Oracle TNFF then fails the RAC communication to the public network.

RAC client HA options

Option 1: Use IPAT or IP Aliasing for the RAC client network

It is recommended to have more than one client network adapter per node. If each RAC node has only one network adapter and it goes down, then that RAC instance can no longer serve any transaction. With two or more adapters for client network per physical node, you can use either the HACMP/ES IP Address Takeover (IPAT) or IP Aliasing to failover to the standby adapter, if the primary adapter fails.

Option 2: Use EtherChannel for the RAC client network

If you plan to use multiple client adapters per pSeries server, you may want to consider bundling all the client network adapters into an EtherChannel. The benefits of using the EtherChannel are that it provides load balancing and also has embedded high availability.

If you have a lot of clients, you may use the EtherChannel hashing algorithm to do the network load balancing.

Multiple physical paths to storage subsystem

Since the database is stored on disks, high availability of storage access should also be considered. Each node should have at least two disk adapters installed and connected to the common disk pool. Additional device drivers may be required to enable this feature (for example, the Subsystem Device Driver used in an ESS environment).

Network switch

The networking infrastructure must be based on a switched configuration. The network switches must “speak the same language” as the pSeries servers to be able to implement the EtherChannel and other communication features.

5.3.4 GPFS planning

In a non-SP environment, GPFS can be used in the following two cluster types:

- GPFS in an HACMP cluster

  For GPFS on an HACMP environment, all disks must be attached to all nodes in the cluster (nodeset). Note that GPFS does *not* use the Concurrent LVM (CLVM) to access the disks.

  It is also important to note that the GPFS cluster built on top of the HACMP environment does not use any of the HACMP/ES high availability features. HACMP/ES only provides the operating environment that GPFS requires for the RSCT subsystems (cluster node membership). Once GPFS is configured, there is no interaction between the subsystems of the HACMP/ES cluster group and the GPFS subsystem.

  Since HACMP is a prerequisite of Oracle9*i* RAC, then if simplified system administration is preferred, we recommend to build GPFS based on the HACMP cluster configuration.

  However, with this option, you need to be careful with the GPFS network adapters’ naming, to avoid the GPFS network being used by RAC as an interconnect (for performance and availability considerations). We recommend that you configure the GPFS network as a “public” network, and the GPFS network adapter named in

alphanumeric order, after all names of the adapters used for RAC interconnects (to avoid its selection in case private interconnect networks fail).

- GPFS in an RSCT Peer Domain (RPD) cluster

  The RSCT subsystem provided by AIX 5L enables you to configure a GPFS cluster in an RPD environment. The RSCT subsystem of AIX 5L removes the existing HACMP prerequisites in the non-PSSP environment.

  The only requirement is that all nodes must have concurrent access to the disks. Please note that the Concurrent LVM (CLVM) functionality of AIX is NOT required in this configuration.

Dedicated fixed and separated GPFS networks

The physical port for a particular IP address must be a fixed piece of hardware that is translated to a fixed network adapter, and is monitored for failure. Topology services should be configured to send and receive heartbeats over this invariant address. This means that *no* other cluster software or client traffic should interfere with this network.

Since the GPFS network is only used for GPFS token manager traffic, in an Oracle9*i* RAC cluster, one dedicated 100 Mbps Ethernet should be enough for GPFS.

Redundant GPFS networks for high availability

When the GPFS network fails on a node, Oracle fails the instance on that node, due to loss of access from that instance to its database and binary files. To eliminate the network as a single point of failure, the GPFS network should also be redundant. We recommend the use of EtherChannel with two 100 Mbit Ethernet adapters installed to provide redundancy.

Note: The GPFS development team is considering the EtherChannel issue, but the EtherChannel support for GPFS may not be officially available by the time this book is published.

GPFS file systems

Depending on the physical storage configuration and limitations, it is possible to have multiple GPFS shared file systems for the RAC implementation. In our test environment we used one GPFS file system for storing Oracle binaries, and another file system for Oracle data files.

GPFS quorum

For all GPFS nodesets consisting of three or more nodes, a GPFS quorum is defined as one plus half of the number of nodes in the GPFS nodeset (referred to as a multinode quorum). In a multinode environment, if the number of the remaining nodes is less than the quorum requirement (50% +1 of the total number of nodes in the GPFS nodeset), you will not be able to perform any GPFS operations.

In an ESS environment with a two-node nodeset, the ESS virtual path (vpath) feature is not supported under single node quorum configuration. This implies that if the only data path to ESS becomes unavailable, the corresponding Oracle instance needs to be aborted.

To solve this problem, we recommend a third GPFS node to serve as the “quorum buster”. You can either include a third node in the cluster, or you can “carve out” a small LPAR from a large pSeries server, such as p690, to serve as quorum buster. A minimum of three nodes in a GPFS cluster is recommended.

GPFS block size

The size of the GPFS data blocks must be 16 KB, 64 KB, 256 KB, 512 KB, or 1 MB. For Oracle9*i* RAC database implementation in an OLTP environment, we recommend the use of a 16 KB GPFS block size. This is to match the Oracle 16 KB DB_BLOCK_SIZE. For a data warehouse environment, it may be worthwhile considering 64 KB for the GPFS block size.

We recommend that you deploy your application in a test environment and find the best block size match for your database type. Always check the latest Release notes for the software you plan to use in your environment.

GPFS pagepool

Oracle does not use the GPFS pagepool (file system cache buffer) at all. This is because Oracle opens GPFS with Direct I/O, which bypasses the GPFS pagepool. Direct I/O avoids the unnecessary overhead for data to be first moved from disk to the file buffer cache and from there to the application buffer.

However, for any non-Direct I/O related to GPFS usage, GPFS still uses pagepool, and it needs to be sized and tuned. In our test environment, we allocated a 128 MB pagepool for our Oracle binaries (stored on GPFS).

5.3.5 Oracle planning

Oracle user ID and group

You can create any AIX ID for Oracle usage. The only two requirements are that this AIX ID must be in the DBA group that you specified at Oracle installation, and it must be included in the *hagsuser* group (which you also need to create).

Oracle user ID home directory

The Oracle user ID home directory is normally used for storing the shared initialization and tuning parameters, database creation scripts, and applications. It can be placed in:

- Local JFS or JFS2 file systems on each node

  Since the Oracle user ID has a separate home directory on each node, you need to periodically synchronize the scripts, parameter files, and applications.

- An external NFS file system

  For high availability reasons, the file system should be allocated in an external file server node, and then NFS-mounted or automounted on the RAC node. Since this NFS-mounted Oracle user ID home directory offers the same view across all RAC nodes, it offers the most straightforward way of managing the Oracle user ID home directory (for example, the PSSP user management feature).

- A separate GPFS system (apart from the database GPFS file systems).

  It is not recommended to allocate the Oracle user ID home directory in the same GPFS file system used for storing data files.

Oracle binaries

The Oracle binaries can be installed in:

- A local JFS or JFS2 file system on each node

  Installing the Oracle binaries on each node takes more time for installation. The benefit of this option is that it provides better availability.

- One shared GPFS file system

  If you plan to install the Oracle binaries in a GPFS file system, we recommend to use a separate GPFS file system.

  The main advantage of storing the Oracle binaries in a shared GPFS file system is that Oracle installation does not need to copy the code on all nodes in the cluster (as with the previous option). The installed binary files are shared among all the Oracle instances.

  Because of the large number of files in the Oracle binaries file system, it is necessary to increase the number of inodes for the GPFS file system beyond the default value (use the **df** command to check the inode count and percent used).

  You can also use the following command to increase the number of inodes:

  ```
  mmchfs -F MaxNumInodes
  ```

  Our Oracle9i home contains about 64,000 files.

Oracle9*i* RAC installation requires the HACMP cluster to be started

The Oracle cluster manager provides a global view of the cluster and all the nodes in it. It also controls the cluster membership. The cluster manager is typically vendor-supplied. In a pSeries environment, the cluster manager role is provided by the HACMP/ES (through RSCT group services and the topology services).

Note: In an AIX environment, Oracle uses internal functions to identify the cluster type it has to install on. The only supported cluster types are HACMP/ES and PSSP.

Before starting the RAC installation, you need to bring up the HACMP/ES cluster. If the HACMP is not up, Oracle Universal Installer (OUI) will not recognize the cluster and will not make available the RAC binaries for installation. The install will proceed to the end, and none of the RAC-related components will be installed.

To make sure the Oracle9i RAC is properly installed, you should check the following:

- Before the Oracle9*i* RAC installation, check the HACMP cluster status with the command:

  ```
  lssrc -a |egrep "ES|svcs"
  ```

- When you execute **rootpre.sh**, you should see the following message:

  ```
  Adding r/w permissions for group hagsuser to /var/ha/soc/grpsvcsdsocket.<cluster_ame>..
  success.
  ```

  If you do not see this message, you should verify your HACMP cluster status again.

- If the Oracle9i RAC installation is successful, you should find the **srvcctl**, **srvconfig**, **gsd**, and **gsdctl** commands in the $ORACLE_HOME/bin directory.

Oracle instance naming

RAC instances must have unique names. An instance name should be assigned at the operating system level, using the ORACLE_SID environment variable. Test your Oracle user environment (~/.profile) to verify that ORACLE_SID on each node is unique. It is recommended that you use the SIDs that consist of the database name as the common base, and the thread ID of the node's instance (<DB_NAME><THREAD_ID>).

srvctl shared file

The Server Control Utility (srvctl) shared file is used to administer the RAC database. The srvctl shared file should be located on the GPFS file system (this is a shared file).

5.3.6 Memory planning

In this section we provide some simple guidelines to find out how much memory can be used by Oracle without involving any paging activity.

Identify the hardware memory installed

To find out the total physical memory available on your machine, see Example 5-1:

Example 5-1 Commands to identify system memory

```
# rmss -p
# bootinfo -r
# lsattr -El sys0 | grep realmem
```

Identify the system memory used

The **svmon -G** command can be used to identify the *free* memory. The *free* field reports the number of 4 KB frames free in all memory pools.

Oracle opens GPFS with Direct I/O - no pagepool requirement

Since Oracle opens GPFS with Direct I/O, it bypasses the GPFS pagepool for all database file operations, including the Recovery Manager (RMAN) utility. There is no need to size the GPFS pagepool for database usage.

Size pagepool for non-Direct I/O-related GPFS usage

However, you do need to size and tune the pagepool if you:

- Install Oracle binaries on a separate GPFS file system.
- Use other GPFS file systems to store database backups, exports, etc. (using system commands such as **cp** or **tar**).

In our test environment, we allocated 128 MB GPFS as pagepool for the GPFS file system that holds the Oracle binaries.

Memory sizing requirement for other applications

If you plan to run additional applications or tools on the same RAC servers, you need to size the memory requirements for each application.

Estimate Oracle usable memory

The goal is to estimate the amount of memory that Oracle can allocate. Overcommitting system memory (RAM) can result in paging activity, which can cause severe performance degradation. You may use the following guidelines to estimate the Oracle usable memory:

- Available memory =
  {(Hardware Memory) - (System Memory used) - (GPFS pagepool size) - (miscellaneous)}
- Reserve 20% of the available memory for further Oracle tuning or adjustment.
- Use 80% of the available memory as a target; start your Oracle instance with
  (SGA_MAX_SIZE + PGA_AGGREGATE_TARGET)
  which should be less than the memory target.

  Note that in an OLTP environment, PGA does not need to be very large, but a larger SGA normally helps Oracle performance, because more data and index can be cached.

  In a data warehouse environment, the SGA is typically small, due to low data reuse (cache hit is poor). Most of the memory should be allocated for Program Global Area (PGA), such as SORT_AREA_SIZE and HASH_AREA_SIZE.

5.3.7 Storage planning

Size the database storage required

To determine the database storage requirement, you need to estimate the following:

- The software executables of Oracle and any application running in the same environment.
- The physical database size, including tablespaces for tables and indexes, redo log files, undo tablespaces, and archived redo log files.
- The staging area, including space for flat files waiting to be loaded (installation code, backups, etc.).
- The export area, including space for database exports and other DBA tasks.

You also need to consider future growth.

Direct-attached RAID or JBOD

Redundant Array of Independent (Inexpensive) Disks (RAID) technology is widely adopted as an enhancer for both data integrity and performance. It allows a set of disk drives to be managed as a group. The RAID configuration can survive to some degrees of disk failure.

Although GPFS allows you to specify how many copies of data to maintain, it is not recommended to use “Just a Bunch of Disks” (JBOD - physical disks (LUNs) configured in a 1:1 relationship with system logical disks).

GPFS file system replication ensures that the latest critical data updates are preserved in the event of a disk failure. From a performance standpoint, in certain configurations, hardware RAID could perform better than the GPFS replication.

A very large RAID array trades off high availability

The disk failure protection of a very large RAID array, for example 15+P, is not as high as a small RAID array, for example 4+P. From the cost point of view, a large RAID array is apparently more efficient, but, under multiple disk failure, it cannot offer the same disk protection as a smaller RAID array.

RAID-5 or RAID-10

The most commonly used RAID levels in a database environment are RAID-5 or RAID-10. RAID-5 spreads read and write data and parity across all disks. The parity segments are rotated among the drives. The RAID-5 enhances sequential read performance, but decreases overall write performance. RAID-10 (also known as RAID 0+1) does striping and mirroring. Considering cost, RAID-5 is more effective than RAID-10.

Enterprise Storage Server (ESS)

The ESS system is made up of two 4-way or 8-way 64-bit SMP clusters. Each cluster can have its own cache and non-volatile storage. Each RAID array can be configured as RAID-5 or RAID-10. See the following ESS RAID guidelines for database usage.

ESS RAID-5 can be used for heavy update

Historically, due to the classic RAID-5 “write penalty”, database vendors and consultants do not recommend the use of RAID-5 for heavy write activities. However, if ESS is used, this “write penalty” does not affect application performance due to the large cache and 100% cache write hits.

This means that ESS returns a “write complete” status to the server as soon as the data block has been written to cache and to Non-Volatile Storage (NVS). Therefore, as long as the

backend (physical) disk configuration can support the sustained I/O rate required by the application, write I/O performance is not determined by the RAID-5 or RAID-10 choice.

For more information, see the IBM EAS Technical Brief, "Configuring the Enterprise Storage Server (ESS) for Oracle OLTP Applications", by Dale Martin.

ESS RAID-5 can be used for Oracle redo logs

Another “classic” Oracle recommendation is not to use RAID-5 devices for RDBMS log files, because the RAID-5 design “cannot support high sequential write activity.”

The ESS (as in all other IBM hardware RAID-5 capable storage offerings) uses a “full stripe” parity generation algorithm that eliminates the need to read existing data or parity in order to generate the new parity information.

Whereas a classic RAID-5 implementation may require four physical I/Os per logical I/O write request, the ESS “full stripe” RAID-5 write requires at most 1.17 physical I/Os per logical I/O write request (one parity write for every six data writes).

For high volume sequential write applications, this is an advantage for RAID-5 over RAID-10, which may require two physical writes (one to each copy) per logical write request. (See the IBM EAS Technical Brief, "Configuring the Enterprise Storage Server (ESS) for Oracle OLTP Applications", by Dale Martin).

Use RAID-10 if the random write percentage is high

For most database workloads, RAID-5 provides comparable performance to RAID-10, with considerably lower cost per megabyte. This makes RAID-5 the default choice for most applications.

RAID-10 should be considered for workloads with a high percentage of random write activity and high I/O access densities. There is no “one size fits all” answer, but a basic rule-of-thumb would be to consider using RAID-10 if the random write content exceeds 25% of the overall write activities, and the peak sustained I/O rate is expected to exceed 50% of the array capacity. (See the IBM EAS Technical Brief, "Configuring the Enterprise Storage Server (ESS) for Oracle OLTP Applications", by Dale Martin).

Do not allocate too many LUNs for each ESS RAID array

Allocating too many LUNs per RAID array may cause the ESS array to be flooded with I/O requests under heavy I/O workloads. This may create some performance problems for GPFS.

FAStT Storage server

FAStT Storage server architecture is based on Fibre Channel architecture, inside and outside (including FC disks). It can be configured in RAID-0, RAID-1, RAID-3, RAID-5, and RAID-10 modes, and you can have a mixture of RAID levels using multiple arrays inside a single server.

Turn off write cache mirroring for write-most workload

For heavy write workloads, we recommend to turn off the FAStT mirrored write cache feature. This is because the mirrored writes between the two controller caches use the internal FC bus to replicate. You can still use the write cache to protect the data integrity.

5.4 Physical database design

The proper layout of datafiles is critical for database operations. A good physical database design should eliminate all I/O hot spots. The GPFS file system offers a striping I/O infrastructure, useful for accomplishing this goal.

5.4.1 Oracle Striped and Mirrored Everything (SAME) strategy

Configuring storage subsystems for the Oracle database can be a very complex process. A classic physical database design requires the physical separation of redo logs from data, tables from indexes, and separation of rollback segments. In many cases, I/O performance problems can be traced to “hot” files or physical disks. When a bottleneck occurs, system administrators or database administrators need to manually relocate the high activity data files.

Oracle recommendations are for a simple, efficient, and highly available storage configuration, called Striped and Mirrored Everything (SAME) architecture (see the Oracle whitepaper, “Optimal Storage Configuration Made Easy”, by Juan Loaiza, Oracle Corporation). The basic idea of this configuration is to make extensive use of striping across large sets of disks.

To achieve high availability, the disks should be mirrored. This SAME configuration, as recommended by Oracle, produces close to optimal performance for all types of workloads, OTLP, data warehouse, and batch processing.

The disadvantages of this recommendation come from the fact that because of the large capacities of the current disks, a lot of disk space may remain unused, or because this architecture does not consider the internal design (RAID, cache, bus) of the modern storage subsystems.

5.4.2 GPFS architecture is based on a similar concept as Oracle SAME

GPFS is in line with the SAME physical database layout strategy. The GPFS I/O operations are striped across all the available disks, according to the following rules:

- One physical volume per RAID logical unit or disk.
- One Volume Group per physical volume.
- One logical volume per volume group.
- For RAC, one nodeset per GPFS cluster.
- GPFS I/O is striped across all the available disks based on the GPFS block size.

To ensure no data loss, the disk subsystem should be configured as RAID. To provide high performance, each GPFS file system storing database files should be striped across as many disks as possible.

This enables any Oracle datafile (such as system tablespace, redo logs, or different types of tablespaces) to be allocated as a single file under a GPFS file system (thus making DB administrative tasks easy).

The RAC implementation using GPFS can greatly simplify the RAC physical database layout for both OLTP or data warehouse databases, and eliminate the I/O hot spots.

5.5 RAC basic implementation steps

1. Hardware preparation

   Install pSeries servers and networks; all disks multi-tailed to all the RAC nodes.

2. Build the HACMP/ES cluster, and test all the high availability features you configured. For example, test the IPAT or IP aliasing of the client network if you have one. It is highly recommended that you do a thorough test of the HACMP cluster before proceeding to the next step.

3. Build the GPFS cluster and file systems to accommodate the database and Oracle executables.

4. Create an *oracle* user ID with the dba group and hagsuser group. Set up the user's "~/.profile" file with all Oracle-required environment variables and PATH settings.

5. Install the Oracle binary into either a separated GPFS file system apart from the database GPFS file systems, or into a local JFS/JFS2 of each RAC node. Note that you need to bring up the HACMP cluster before the Oracle installation.

6. Prepare the database creation scripts and initialization parameters. These scripts should include database creation, tablespace creation, and the initialization parameters files, tnsnames.ora and listener.ora. These database creation scripts can be either generated by Oracle'**dbca**, **netca**, or manually created. Make sure each instance has its own thread of redo logs and undo tablespaces.

7. Execute the database creation scripts from one node, preferably from the first RAC node. If your scripts are generated by **dbca**, you need to start up the listener first.

8. After the database creation, make sure that all the init.ora, SPFILE, listener.ora and tnsnames.ora files are properly in place. Test the startup and shutdown of each Oracle instance. If srvctl is used, make sure the GSD is up and the srvctl shared file is initialized.

9. Start up the listener on each instance. Test the client connections and Transparent Application Failover.

5.6 RAC client side failover and load balancing

A properly configured RAC should not only provide the server-side high availability, it should also take into consideration the client connection failover. To complete the high availability picture, you need to be familiar with both the connect time failover and runtime Transparent Application Failover and how it is configured.

This section describes some of the Oracle client failover features, in case of an instance failure in a Real Application Cluster environment. A sample of the listener.ora and tnsnames.ora setup and its test results can be found in Chapter 6, "High availability test scenarios" on page 169.

Client side connect time failover

When more than one listener supports a service, a client can be configured to failover the client request to a different listener if the first listener fails. Reconnection attempts continue until the client successfully connects to a listener.

Transparent Application Failover, a runtime failover

The Oracle Transparent Application Failover (TAF) feature is a runtime failover for the high availability environment, such as the RAC environment. TAF can failover and establish the application connections. It enables client applications to automatically reconnect to the

database if the connection fails, and, optionally, resume a “select” statement, that was in progress. The reconnection happens automatically from the Oracle Call Interface library.

Client side connect time load balance

When more than one listener supports a service, a client can randomize requests to various listeners. This feature distributes the load so as not to overburden a single listener. Without client side connect time balancing, Oracle Net Service will try each of the protocol addresses sequentially until one succeeds.

5.7 References

1. Oracle9i Release Notes, Release 2 (9.2.0.1.0) for AIX-Based 5L Systems
2. Configuring the IBM General Parallel File System (GPFS) in the Oracle Real Application Cluster (RAC) Environment, by Rick Piasecki
3. Oracle white paper: “Optimal Storage Configuration Made Easy”, by Juan Loaiza, Oracle Corporation
4. Implementing Oracle9i RAC with Linux on IBM @server xSeries® Servers
5. “IBM’s General Parallel File System (GPFS) 1.4 for AIX, Architecture and Performance”, by Dominique Heger and Gautam Shah
6. IBM EAS Technical Brief, Configuring the Enterprise Storage Server (ESS) for Oracle OLTP Applications, by Dale Martin
7. Quick Installation Guide Oracle9*i* RAC on IBM @server pSeries with AIX 4.3.3, by Fabienne Lepetit and Michel Passet
8. Configuring Oracle9*i* Real Application Clusters for High Availability on HACMP and AIX, by Ronald M. Bautista
9. Step by Step Installation of RAC on IBM AIX (RS/6000), Oracle Metalink white paper
10. Oracle9*i* Real Application Clusters Administration
11. Oracle9*i* Real Application Clusters Concepts
12. GPFS AIX Cluster Administration and Programming Reference
13. GPFS AIX Clusters Concepts, Planning, and Installation Guide

6

High availability test scenarios

This chapter provides the results for some simple tests that can be performed in order to verify the high availability features offered by Oracle9*i* RAC. This includes connect-time load balancing and failover, as well as Transparent Application Failover (TAF).

Those features have to be properly configured in order to produce the correct results. This chapter discusses detailed configurations that were used to obtain the results reported.

We consider the failure of Oracle9*i* RAC-related components such as listener, database instance, interconnect network interface, and client network interface, which have a direct impact on the availability of the database.

We also consider the effect of failures of platform components, which are not directly known to the Oracle9*i* RAC software, but which are relied upon when providing services to the clients. These components include the GPFS subsystem and the network used by this subsystem to provide the cluster file system for datafiles and Oracle9*i* executable files, as well as the server as a whole.

As always, there may be different ways to achieve the same result, and we do not claim that the test solutions offered in this chapter are the only ones that will work in a similar environment. But we feel that they can serve as a good starting point for adapting to the requirements of your specific situation.

6.1 Test objectives and procedure

This section describes the test procedure we used in our environment to verify the availability features of the Oracle9*i* RAC database, as well as of some critical components in our configuration.

For our test we used two different types of sqlplus (client) sessions:

- Idle session - We print the timestamp and the instance name before and after the failure. No session activity takes place at the time the failure occurs. We want to verify that reconnect to another instance occurs if new queries are issued from the same session after the failure.
- Active session - We print the timestamp and instance name before and after the failure. After the first timestamp we issue a **select** statement which runs for at least one minute, and during this interval we initiate the failure condition. We want to verify that the running query is not affected by the failure, and a failover to another instance occurs during the processing of the query.

We use a test script to automate the procedure, and to provide a protocol of the steps taken, including timestamps and the results of the query operations.

6.1.1 Client setup

For demonstrating our test case we implement a special OracleNet configuration by editing the tnsnames.ora file on the client machine.

On the client we add the entries shown in Example 6-1 to the tnsnames.ora file in order to enable the connect time load balancing and failover, as well as the Transparent Application Failover (TAF) features for connections to the database named rac.

Example 6-1 Entries from the tnsnames.ora file on the client

```
# --- 1st entry: load balancing, connect time failover, TAF method=basic

RAC =
  (DESCRIPTION =
    (enable=broken)
    (LOAD_BALANCE=ON)
    (FAILOVER=ON)
      (ADDRESS = (PROTOCOL = TCP)(HOST = node1)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node2)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node3)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node4)(PORT = 1521))

    (CONNECT_DATA =
      (SERVICE_NAME = RAC)
      (failover_mode=(type=select)(method=basic)(retries=20)(delay=5))
    )
  )

# --- 2nd entry: no load balancing, but connect time failover and TAF method=basic

RAC1F =
  (DESCRIPTION =
    (enable=broken)
    (LOAD_BALANCE=NO)
    (FAILOVER=ON)
```

```
      (ADDRESS = (PROTOCOL = TCP)(HOST = node1)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node2)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node3)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node4)(PORT = 1521))

    (CONNECT_DATA =
      (SERVICE_NAME = RAC)
      (failover_mode=(type=select)(method=basic)(retries=20)(delay=5))
    )
  )

RAC2F =
  (DESCRIPTION =
    (enable=broken)
    (LOAD_BALANCE=NO)
    (FAILOVER=ON)
      (ADDRESS = (PROTOCOL = TCP)(HOST = node2)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node3)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node4)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node1)(PORT = 1521))

    (CONNECT_DATA =
      (SERVICE_NAME = RAC)
      (failover_mode=(type=select)(method=basic)(retries=20)(delay=5))
    )
  )

# --- 3rd entry: no load balancing, but connect time failover, TAF method=preconnect

RAC1P =
  (DESCRIPTION =
    (enable=broken)
    (LOAD_BALANCE=NO)
    (FAILOVER=ON)
      (ADDRESS = (PROTOCOL = TCP)(HOST = node1)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node2)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node3)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = node4)(PORT = 1521))

    (CONNECT_DATA =
      (SERVICE_NAME = RAC)
      (failover_mode=(type=select)(method=preconnect)(backup=rac2f))
    )
  )
```

The first entry enables load balancing and connect-time failover among all four instances if the connection is established using the connect string `@rac`. The line `(enable=broken)` enables the client to make use of the TCP “keep alive” mechanism to detect whether a session is to be kept alive or is regarded to be broken. This is used if the server fails or can no longer be reached via the client network.

TAF is also enabled and the basic failover method is used, that is, the session is connected to another instance when the connection to the original instance is lost. This will be retried 20 times with a delay of 5 seconds between retries.

The second entry is used with the connect string `@rac1f`. This entry is the same as the first one, except that the load balancing is switched off. Sessions using this connect string will

therefore be connected to instance rac1, as long as it is available. In case instance rac1 becomes unavailable, all connections are directed to instance rac2.

The last entry enables the preconnect failover method, if connect string `@rac1p` is specified. This establishes two connections at the time the session is started, but only the primary connection is used, as long as it is available. If this connection becomes unavailable, the session switches to the secondary connection. This should provide faster failover than the basic method, since the secondary connection is already established at the time the failure occurs.

For a symmetrical installation, in which all the instances behave the same way, the corresponding entries for the other instances have to be added to the tnsnames.ora file. Unless otherwise noted, we always use instance rac1 for our tests, so only these entries are specified.

A complete description of the syntax used for the different failover configurations can be found in the Oracle9*i* Net Services Administrator's Guide.

6.1.2 Test query

The script used for running the test query is shown in the Example 6-2.

Example 6-2 Script to run test query

```
#!/bin/ksh

if [[ "$(whoami)" != "npora" ]]
then
   echo I am $(whoami).
   echo please run me as npora...
   exit
fi

cd $TESTDIR

sqlplus pet/pet@$1 << ! > /dev/null
spool ha_test
@time_instance
select count(*) from
( select * from dba_source
  union
  select * from dba_source
  union
  select * from dba_source
  union
  select * from dba_source
  union
  select * from dba_source);
@time_instance
spool off
exit
!
```

This script is used for the active session that contains an SQL **select** statement running for a time long enough to induce a failure condition on the server, while it is running. For the idle session we replace the query by an operating system **sleep** command:

```
!sleep 120
```

If the test query runs uninterrupted, it produces the output in the spool file shown in Example 6-3.

Example 6-3 Spool file of test query running unperturbed

```
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
05-JUN-03 09.56.30.181362 AM -04:00 rac1

SQL> select count(*) from
  2  ( select * from dba_source
  3    union
  4    select * from dba_source
  5    union
  6    select * from dba_source
  7    union
  8    select * from dba_source
  9    union
 10    select * from dba_source);

  COUNT(*)
----------
    116652

SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
05-JUN-03 09.58.09.507318 AM -04:00 rac1

SQL> spool off
```

The execution time of this query is about 100 seconds.

6.1.3 Test script

To automate the test we use the following shell script (see Example 6-4). We adapt it for the different test scenarios by choosing the test query and inserting the appropriate failure condition code and the corresponding repair action code.

The failure condition code is used to simulate the failure of the component that is the subject of the test. After the test, the repair action code is used to repair the damage as far as possible by performing the steps necessary to recover from the failure.

Both additions to the script are explained in more detail in the subsections describing the test scenarios; see 6.2, “Database availability tests” on page 174 and 6.3, “Platform availability tests” on page 191.

Example 6-4 Test script for various failover conditions

```
#!/bin/ksh
```

```
sleeptime=45
connect=$1

if [[ "$(whoami)" != "root" ]]
then
   echo I am $(whoami).
   echo please run me as root ...
   exit
fi

if [[ "$connect" = "" ]]
then
   connect=rac1f
fi

echo "$(date) Test will connect to $connect ..."
echo "$(date) Starting test ..."
echo "$(date) Starting query ..."

#  *** Put your query here:

su - npora "-c $TESTDIR/test_query.sh $connect" &

#  *** End of query code

echo "$(date) Waiting for $sleeptime seconds ..."
sleep $sleeptime

#  *** Put your failure code here ...


#  *** End of failure code

echo "$(date) Waiting for query to finish ..."
wait
echo "$(date) Query finished ..."

cat ha_test.lst

#  ***Put your repair code here ...


#  *** End of repair code

echo "$(date) Test finished ..."
```

6.2 Database availability tests

This section describes some tests to demonstrate the availability features built into Oracle9*i* RAC. We consider the following test cases:

- Listener fails on one node
- Database instance fails on one node
- Interconnect network interface fails on one node
- Client network interface fails on one node

In each of the test cases we observe a session that is connected to the instance on the node which experiences the failures. We record the result of a query that is running at the time of the failure and note the instance we are connected after the failure occurs.

In some of the test cases we perform additional tests, described in more detail in the corresponding subsections.

6.2.1 Listener fails on one node

For this test, we stop the listener on the node where the test query is running. This is accomplished by adding the following failure and repair code to the test script at the places indicated therein (see Example 6-4).

Example 6-5 Failing listener - failure and repair code for test script

```
# --- failing listener test
# --- code to be included in test script
#
# --- failure code:
#
echo “$(date) Shutting down listener ...”
rsh node1 -n “su - oracle \”-c lsnrctl stop\” “ >/dev/null 2>&1
#
# --- repair code:
#
echo “$(date) Starting up listener ...”
rsh node1 -n “su - oracle \”-c lsnrctl start\” “ >/dev/null 2>&1
```

Running the adapted test script produces the output shown in Example 6-6.

Example 6-6 Listener failing for instance processing query - test script output

```
Wed Jun  4 17:44:19 EDT 2003 Test will connect to rac1f ...
Wed Jun  4 17:44:19 EDT 2003 Starting test ...
Wed Jun  4 17:44:19 EDT 2003 Starting query ...
Wed Jun  4 17:44:19 EDT 2003 Waiting for 45 seconds ...
Wed Jun  4 17:45:04 EDT 2003 Shutting down listener ...
Wed Jun  4 17:45:04 EDT 2003 Waiting for query to finish ...
Wed Jun  4 17:45:57 EDT 2003 Query finished ...
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
04-JUN-03 05.44.19.830042 PM -04:00 rac1

SQL> select count(*) from
  2  ( select * from dba_source
  3    union
  4    select * from dba_source
  5    union
  6    select * from dba_source
  7    union
  8    select * from dba_source
  9    union
 10    select * from dba_source);

  COUNT(*)
```

```
----------
    116652

SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
---------------------------------- ----------------
04-JUN-03 05.45.57.463320 PM -04:00 rac1

SQL> spool off
Wed Jun  4 17:45:57 EDT 2003 Starting up listener ...
Wed Jun  4 17:45:57 EDT 2003 Test finished ...
```

As you can see from the listing above, the query is finished with the correct result on the same instance (rac1) it was started. Runtime of the query is also not affected by failing the listener.

This is because the listener is only responsible for initiating new connections to an instance. Once the connection is established, the listener is no longer needed. Existing connections are not affected by a failed listener, but new connections can no longer be established to the instance running on the node without a listener.

6.2.2 Database instance fails on one node

For the case where a database instance fails on one node, we perform two different types of tests:

- Single session - A single session is connected to an instance which is shut down while it is idle or a query is running. This test corresponds to the tests described in the previous section.
- Double session - Two sessions are connected to two different instances. Each session inserts a row into the same table and commits. Then each session updates the row just inserted by the other session. Before they can commit the update, one of the instances fails. We study the effect of the failure on the contents of the table.

Single session tests

For the single session test, we induce a database instance failure while it is running a query.

The following failure and repair code is included in the test script (Example 6-7).

Example 6-7 database instance failure - failure and repair code for test script

```
# --- failing database instance test
# --- code to be included in test script
#
# --- failure code:
#

echo "$(date) Shutting down instance ..."
rsh node1 -n "/home/np/shutdown_abort.sh"

#
# --- repair code:
#

echo "$(date) Starting up instance ..."
```

```
rsh node1 -n "/home/np/startup.sh"
```

The scripts shutdown_abort.sh and startup.sh each contain an sqlplus session connecting as sysdba, and running the **shutdown abort** and **startup** commands respectively.

Idle session

Running the test script with an idle sqlplus session, using the operating system command **sleep 120** as a dummy query, gives the result shown in Example 6-8.

Example 6-8 Failing instance with idle session - test script output

```
Wed Jun  4 15:18:39 EDT 2003 Test will connect to rac1f ...
Wed Jun  4 15:18:39 EDT 2003 Starting test ...
Wed Jun  4 15:18:39 EDT 2003 Starting query ...
Wed Jun  4 15:18:39 EDT 2003 Waiting for 45 seconds ...
Wed Jun  4 15:19:24 EDT 2003 Shutting down instance ...
Wed Jun  4 15:19:24 EDT 2003 Shutting down instance on node1 ...
Wed Jun  4 15:19:38 EDT 2003 Waiting for query to finish ...
Wed Jun  4 15:20:44 EDT 2003 Query finished ...
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
04-JUN-03 03.18.41.526376 PM -04:00 rac1

SQL> !sleep 120

SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
04-JUN-03 03.20.44.592434 PM -04:00 rac2

SQL> spool off
Wed Jun  4 15:20:44 EDT 2003 Starting up instance ...
Wed Jun  4 15:20:44 EDT 2003 Starting up instance on node1 ...
Wed Jun  4 15:21:17 EDT 2003 Test finished ...
```

You can observe from the output that the session is started on instance rac1, and is reconnected after the failure of this instance to the next instance, rac2.

Active session - basic failover method

By using the test query instead of the operating system sleep, we get the result shown in Example 6-9.

Example 6-9 Failing instance with active session (failover method=basic) - test script output

```
Mon Jun  9 16:17:20 EDT 2003 Test will connect to rac1f ...
Mon Jun  9 16:17:20 EDT 2003 Starting test ...
Mon Jun  9 16:17:20 EDT 2003 Starting query ...
Mon Jun  9 16:17:20 EDT 2003 Waiting for 30 seconds ...
Mon Jun  9 16:17:50 EDT 2003 Shutting down instance ...
Mon Jun  9 16:17:50 EDT 2003 Shutting down instance on node1 ...
Mon Jun  9 16:18:07 EDT 2003 Waiting for query to finish ...
Mon Jun  9 16:20:03 EDT 2003 Query finished ...
```

```
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
09-JUN-03 04.17.20.929227 PM -04:00 rac1

SQL> select count(*) from
  2  ( select * from dba_source
  3    union
  4    select * from dba_source
  5    union
  6    select * from dba_source
  7    union
  8    select * from dba_source
  9    union
 10    select * from dba_source);

  COUNT(*)
----------
    116652

SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
09-JUN-03 04.20.03.668949 PM -04:00 rac2

SQL> spool off
Mon Jun  9 16:20:03 EDT 2003 Starting up instance ...
Mon Jun  9 16:20:03 EDT 2003 Starting up instance on node1 ...
Mon Jun  9 16:20:31 EDT 2003 Test finished ...
```

The query is completed successfully, and the session is connected to the second instance, rac2. We measure a runtime of the query as about 163 seconds (compared to 100 seconds without interruption).

Active session - preconnect failover method

Using the preconnect failover method, the result shown in Example 6-10 is obtained.

Example 6-10 Failing instance with active session (failover method=preconnect) - test script output

```
Mon Jun  9 16:47:49 EDT 2003 Test will connect to rac1p ...
Mon Jun  9 16:47:49 EDT 2003 Starting test ...
Mon Jun  9 16:47:49 EDT 2003 Starting query ...
Mon Jun  9 16:47:49 EDT 2003 Waiting for 30 seconds ...
Mon Jun  9 16:48:19 EDT 2003 Shutting down instance ...
Mon Jun  9 16:48:20 EDT 2003 Shutting down instance on node1 ...
Mon Jun  9 16:48:34 EDT 2003 Waiting for query to finish ...
Mon Jun  9 16:50:27 EDT 2003 Query finished ...
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
09-JUN-03 04.47.50.686611 PM -04:00 rac1

SQL> select count(*) from
  2  ( select * from dba_source
  3    union
```

```
  4    select * from dba_source
  5    union
  6    select * from dba_source
  7    union
  8    select * from dba_source
  9    union
 10    select * from dba_source);

  COUNT(*)
----------
    116652

SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
09-JUN-03 04.50.27.411402 PM -04:00 rac2

SQL> spool off
Mon Jun  9 16:50:27 EDT 2003 Starting up instance ...
Mon Jun  9 16:50:27 EDT 2003 Starting up instance on node1 ...
Mon Jun  9 16:51:39 EDT 2003 Test finished ...
```

As the script file output demonstrates, the session does not report any errors, but the query starts on instance rac1 and finishes with the correct result on the backup instance rac2 (as designated in the tnsnames.ora file; see Example 6-1 on page 170).

The runtime for the query is about 157 seconds (compared to about 100 seconds for the uninterrupted run, and about 163 seconds for the basic failover method).

Double session test

We create a table called CRASH with columns ID, UPDATED_BY, INSERTED_BY.

In the first transaction, two sessions connected to different instances each insert a row into the table and commit. Before the commit, each session can only select its own rows. After the commit, both sessions can select both rows.

Next, each session updates the row inserted by the other session. After the insertion, each session can select only its own updates.

Before the sessions commit their updates, instance rac2 is failed (**`shutdown abort`**).

After the failure, the session connected to the surviving instance (rac1) can still select the rows in the table, but does not see the update performed by the failed instance (rac2), since it has not been committed.

When trying to select the table, the session originally connected to the failed instance receives an error message, indicating that the transaction has to be rolled back.

So, in this case, the first session commits its update, and the second session rolls back the lost transaction. Subsequently, both sessions can select the table, which contains the row updated by instance rac1, but not the row updated by instance rac2. Furthermore, session 2 is now connected to instance rac3 instead of rac2.

Table 6-1 on page 180 summarizes the transactions.

Table 6-1 Transaction flow for a double session test with instance failure

Step	Session 1	Session 2
1	SQL> @../time_instance CURRENT_TIMESTAMP INSTANCE_NAME ---------- ---------- 30-MAY-03 07.55.40.215229 PM -04:00 **rac1**	SQL> @../time_instance CURRENT_TIMESTAMP INSTANCE_NAME ---------- ---------- 30-MAY-03 07.55.31.792816 PM -04:00 **rac2**
2	SQL> insert into crash values (1, '-', 'rac1'); 1 row created.	SQL> insert into crash values (2, '-', 'rac2'); 1 row created.
3	SQL> select * from crash; ID UPDATED_BY INSERT_BY ---------- ---------- ---------- 1 - rac1	SQL> select * from crash; ID UPDATED_BY INSERT_BY ---------- ---------- ---------- 2 - rac2
4	SQL> commit; Commit complete.	SQL> commit; Commit complete.
5	SQL> select * from crash; ID UPDATED_BY INSERT_BY ---------- ---------- ---------- 2 - rac2 1 - rac1	SQL> select * from crash; ID UPDATED_BY INSERT_BY ---------- ---------- ---------- 2 - rac2 1 - rac1
6	SQL> update crash set UPDATED_BY='rac1' where ID=2; 1 row updated.	SQL> update crash set UPDATED_BY='rac2' where ID=1; 1 row updated.
7	SQL> select * from crash; ID UPDATED_BY INSERT_BY ---------- ---------- ---------- 2 rac1 rac2 1 - rac1	SQL> select * from crash; ID UPDATED_BY INSERT_BY ---------- ---------- ---------- 2 - rac2 1 rac2 rac1
8	instance rac2 is failed by issuing **shutdown abort** from another **sqlplus** session	
9	SQL> select * from crash; ID UPDATED_BY INSERT_BY ---------- ---------- ---------- 2 rac1 rac2 1 - rac1	SQL> select * from crash; select * from crash * ERROR at line 1: ORA-25402: transaction must roll back
10	SQL> commit; Commit complete.	SQL> roll back Rollback complete.

Step	Session 1	Session 2
11	SQL> select * from crash; ID UPDATED_BY INSERT_BY ---------- ---------- ---------- 2 rac1 rac2 1 - rac1	SQL> select * from crash; ID UPDATED_BY INSERT_BY ---------- ---------- ---------- 2 rac1 rac2 1 - rac1
12		SQL> @../time_instance CURRENT_TIMESTAMP INSTANCE_NAME --------------------------------- ---------- 30-MAY-03 08.09.59.691001 PM -04:00 **rac3**

6.2.3 Interconnect network interface fails on one node

In this section we explore some failure conditions involving the cluster interconnect that is used by Oracle9*i* RAC for inter-instance communication. We describe the following features:

- Interconnect failover - If a network interface associated with the cluster interconnect goes down, traffic switches to the secondary interconnect.
- Session behavior - If a network interface associated with the cluster interconnect goes down on one node, sessions connected to the instance running on that node are not affected by the failing interface and the corresponding switch to the secondary network.
- Interconnect load balancing - We configure the interconnect for load balancing over two networks. Failing the interface of one of the cluster interconnect networks on one node results in the instance on this node being shut down automatically. There is no high availability feature available if the cluster interconnect is configured for load balancing.

Interconnect failover

For this test we apply some load on all instances, in order to generate traffic over the cluster interconnect. In our configuration, the first network used for the interconnect uses interface en1, the second network uses interface en2. Both networks are labeled as “private” in the HACMP configuration. The client network is a “public” network and uses interface en0.

With all interfaces up and some traffic over the interconnect, we observe the network load using **topas**, as shown in Example 6-11.

Example 6-11 Snapshot of network load - all networks up

```
Network  KBPS   I-Pack  O-Pack   KB-In  KB-Out
en1     232.8      686     672   610.0   554.0
en0      78.2     1929    1318   192.0   199.0
en3      51.6      423     457    54.0   204.0
lo0       2.8      127     127     7.0     7.0
en2       1.0       15      17     2.0     3.0
```

After shutting down the primary interconnect network (interface en1), the interconnect traffic switches to the secondary interconnect network (interface en2), as shown in Example 6-12 on page 182.

Example 6-12 Snapshot of network load - primary private network down

```
Network  KBPS   I-Pack  O-Pack   KB-In  KB-Out
en2     344.2      945    1073   778.0   943.0
en0      89.0     2198    1555   221.0   224.0
en3      89.0     1052    1098   138.0   307.0
lo0       2.8      133     133     7.0     7.0
en1       0.0        0       0     0.0     0.0
```

After shutting down the secondary interconnect network (interface en2), the interconnect traffic switches to the public network interface en0; see Example 6-13.

Example 6-13 Snapshot of network load - all private networks down

```
Network  KBPS   I-Pack  O-Pack   KB-In  KB-Out
en0     295.4     1912    1662   648.1   829.1
en3      59.4      574     612    73.0   224.0
lo0       1.8       92      92     5.0     4.0
en1       0.0        0       0     0.0     0.0
en2       0.0        0       0     0.0     0.0
```

If we now restore the primary interconnect network interface en1, the interconnect traffic reverts back to this network, see Example 6-14.

Example 6-14 Snapshot of network load - secondary private network down

```
Network  KBPS   I-Pack  O-Pack   KB-In  KB-Out
en1     216.2      578     607   522.0   559.0
en3      58.0      589     624    75.0   215.0
en0      55.6     1340     891   136.0   142.0
lo0       1.2       67      67     3.0     3.0
en2       0.0        0       0     0.0     0.0
```

Bringing up the secondary interconnect network interface en2 does not change the overall picture, since the primary interconnect network (interface en1) is still available, see Example 6-15.

Example 6-15 Snapshot of network load - all networks up (after test)

```
Network  KBPS   I-Pack  O-Pack   KB-In  KB-Out
en1     205.2      544     622   463.0   563.0
en0      65.4     1552    1031   159.0   168.0
en3      49.4      442     460    59.0   188.0
lo0      14.0      429     429    35.0    35.0
en2       1.0       19      18     3.0     2.0
```

Session behavior during interconnect failure

To demonstrate that a client session running a query is not affected by the interconnect failover, we again use our test script (see Example 6-4 on page 173).

We use a table named DATA that contains 5.2 million rows, and perform a non-parallel table scan that lasts about 100 seconds. During this time the data is read from the datafiles located on a GPFS file system and the instances exchange information over the primary interconnect network.

The failure and repair codes to be included in the test script are as shown in Example 6-16.

Example 6-16 Failure and repair codes to be included in the test script

```
# --- failing primary interconnect interface test
# --- code to be included in test script
#
# --- failure code:
#

echo “$(date) Shutting down primary interconnect ...”
rsh node1 -n “echo \”$(date) $(chdev -l en1 -a state=down) ...\” ”

#
# --- repair code:
#

echo “$(date) Starting up primary interconnect ...”
rsh node1 -n “echo \”$(date) $(chdev -l en1 -a state=up) ...\” ”
```

Running the adapted test script produces the result shown in Example 6-17.

Example 6-17 Adapted test script results

```
Tue Jun 10 15:40:14 EDT 2003 Test will connect to rac1f ...
Tue Jun 10 15:40:14 EDT 2003 Starting test ...
Tue Jun 10 15:40:14 EDT 2003 Starting query ...
Tue Jun 10 15:40:14 EDT 2003 Waiting for 30 seconds ...
Tue Jun 10 15:40:44 EDT 2003 Shutting down primary interconnect ...
Tue Jun 10 15:40:44 EDT 2003 en1 changed ...
Tue Jun 10 15:40:44 EDT 2003 Waiting for query to finish ...
Tue Jun 10 15:42:18 EDT 2003 Query finished ...
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
10-JUN-03 03.40.14.832135 PM -04:00 rac1

SQL> select /*+ noparallel */ count(*) from data;

  COUNT(*)
----------
   5200000

SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
10-JUN-03 03.42.18.243060 PM -04:00 rac1

SQL> spool off
Tue Jun 10 15:42:18 EDT 2003 Starting up primary interconnect ...
```

```
Tue Jun 10 15:42:18 EDT 2003 en1 changed ...
Tue Jun 10 15:42:18 EDT 2003 Test finished ...
```

As you can see from the results, the query is finished on the same instance it was started and gives the correct result. The runtime difference from the uninterrupted run is only about 23 seconds, since the failover of the interconnect is a matter of a few seconds only.

Interconnect load balancing

In order to configure the cluster interconnect for load balancing over two networks, we add the lines shown in Example 6-18 to the init.ora file for all instances.

Example 6-18 Entries in init.ora to configure load balancing on cluster interconnect networks

```
rac1.cluster_interconnects=10.10.10.61:172.16.10.61
rac2.cluster_interconnects=10.10.10.62:172.16.10.62
rac3.cluster_interconnects=10.10.10.63:172.16.10.63
rac4.cluster_interconnects=10.10.10.64:172.16.10.64
```

Note: IP addresses for the networks have to appear in the same order for all instances.

To demonstrate the load balancing feature, we again apply some load on all nodes and observe the networks using **topas** (Example 6-19). Interfaces en1 and en2 are associated with the IP-addresses from the CLUSTER_INTERCONNECTS parameter.

Example 6-19 Snapshot of network load - load balancing on interconnect

```
Network  KBPS   I-Pack  O-Pack   KB-In  KB-Out
en2     128.8     395     359   356.0   288.0
en1     114.2     307     289   303.0   268.0
en0      69.2    1666     900   172.0   174.0
en3      52.0     550     542   172.0    88.0
lo0       0.8      17      17     2.0     2.0
```

After we shut down interface en1 on this node, the interconnect traffic disappears (Example 6-20).

Example 6-20 Snapshot of network load - interconnect down after interface failure

```
Network  KBPS   I-Pack  O-Pack   KB-In  KB-Out
en3       5.2      44      44    20.0     6.0
en2       2.0      43      16     8.0     2.0
en0       1.0      26      23     2.0     3.0
lo0       0.4      16      16     1.0     1.0
en1       0.0       0       0     0.0     0.0
```

The remaining packets going over en2 are due to HACMP heartbeat. If we again bring up the interface before the failure is detected by Oracle9*i*, operations are resumed and no errors are reported. If the interface remains down for several minutes after the initial failure, some messages appear in the alert log of instance rac1(Example 6-21 on page 185).

Example 6-21 Entries from alert log during interconnect failure

```
Thu Jun 12 10:18:03 2003
Communications reconfiguration: instance 1
Communications reconfiguration: instance 3
Thu Jun 12 10:18:04 2003
Trace dumping is performing id=[cdmp_20030612101803]
Thu Jun 12 10:18:08 2003
Communications reconfiguration: instance 2
Thu Jun 12 10:18:34 2003
Trace dumping is performing id=[cdmp_20030612101804]
Thu Jun 12 10:18:38 2003
Trace dumping is performing id=[cdmp_20030612101847]
Thu Jun 12 10:19:49 2003
Waiting for clusterware split-brain resolution
Thu Jun 12 10:29:49 2003
Errors in file /oracle/admin/rac/bdump/rac1_lmon_512040.trc:
ORA-29740: evicted by member 0, group incarnation 5
LMON: terminating instance due to error 29740
Thu Jun 12 10:29:49 2003
Errors in file /oracle/admin/rac/bdump/rac1_dbw0_708646.trc:
ORA-29740: evicted by member , group incarnation
Thu Jun 12 10:29:50 2003
Errors in file /oracle/admin/rac/bdump/rac1_lms1_503948.trc:
ORA-29740: evicted by member , group incarnation
Thu Jun 12 10:29:50 2003
Errors in file /oracle/admin/rac/bdump/rac1_lgwr_1159242.trc:
ORA-29740: evicted by member , group incarnation
Thu Jun 12 10:29:50 2003
Errors in file /oracle/admin/rac/bdump/rac1_lck0_1192010.trc:
ORA-29740: evicted by member , group incarnation
Thu Jun 12 10:29:52 2003
Errors in file /oracle/admin/rac/bdump/rac1_pmon_610538.trc:
ORA-29740: evicted by member , group incarnation
Instance terminated by LMON, pid = 512040
```

The effect of the interface failure is to shut down the instance on the node that lost its interface. By configuring the interconnect for increased bandwidth by enabling load balancing, we lose the high availability features of the interconnect.

6.2.4 Client network interface fails on one node

For this test we shut down the client network interface on the node running the instance that processes the test query.

The failure condition code and the repair code used in the test script are shown in Example 6-22.

Example 6-22 Failing client network interface - failure and repair code for test script

```
# --- failing client network interface test
# --- code to be included in test script
#
# --- failure code:
#

echo "$(date) Shutting down interface ..."
```

```
rsh node2 “echo \”$(date) \$(rsh onet21 chdev -l en0 -a state=down) ...\” “

#
# --- repair code:
#

echo “$(date) Starting up interface ...”
rsh node2 “echo \”$(date) \$(rsh onet21 chdev -l en0 -a state=up) ...\” “
```

Idle session

With the idle session (operating system **sleep 120** instead of the actual query) the test script produces the output shown in Example 6-23.

Example 6-23 Failing interface with idle session - test script output

```
Thu Jun  5 11:29:30 EDT 2003 Test will connect to rac1f ...
Thu Jun  5 11:29:30 EDT 2003 Starting test ...
Thu Jun  5 11:29:30 EDT 2003 Starting query ...
Thu Jun  5 11:29:30 EDT 2003 Waiting for 45 seconds ...
Thu Jun  5 11:30:15 EDT 2003 Shutting down interface ...
Thu Jun  5 11:30:15 EDT 2003 en0 changed ...
Thu Jun  5 11:30:16 EDT 2003 Waiting for query to finish ...
Thu Jun  5 11:43:56 EDT 2003 Query finished ...
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
05-JUN-03 11.29.30.845725 AM -04:00 rac1

SQL> !sleep 120

SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
05-JUN-03 11.43.56.724708 AM -04:00 rac2

SQL> spool off
Thu Jun  5 11:43:56 EDT 2003 Starting up interface ...
Thu Jun  5 11:43:56 EDT 2003 en0 changed ...
Thu Jun  5 11:43:57 EDT 2003 Test finished ...
```

According to the test script output, the sqlplus session starts on instance rac1 and is reconnected to instance rac2. The total runtime of the query is longer than 14 minutes, instead of the 2 minutes the **sleep 120** would need. This difference is due to TCP/IP time-out periods.

Active session - basic failover method

Running the test query while the interface is brought down on the node where the instance is processing the query, leads to the result shown in Example 6-24.

Example 6-24 Failing interface with active session (failover=basic) - test script output

```
Thu Jun 12 16:47:31 EDT 2003 Test will connect to rac1f ...
Thu Jun 12 16:47:31 EDT 2003 Starting test ...
```

```
Thu Jun 12 16:47:31 EDT 2003 Starting query ...
Thu Jun 12 16:47:31 EDT 2003 Waiting for 30 seconds ...
Thu Jun 12 16:48:01 EDT 2003 Shutting down interface ...
Thu Jun 12 16:48:01 EDT 2003 en0 changed ...
Thu Jun 12 16:48:01 EDT 2003 Waiting for query to finish ...
Thu Jun 12 17:02:05 EDT 2003 Query finished ...
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
12-JUN-03 04.47.31.311525 PM -04:00 rac1

SQL> select count(*) from
  2  ( select * from dba_source
  3    union
  4    select * from dba_source
  5    union
  6    select * from dba_source
  7    union
  8    select * from dba_source
  9    union
 10    select * from dba_source);
select count(*) from
*
ERROR at line 1:
ORA-03113: end-of-file on communication channel


SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
12-JUN-03 05.02.05.550570 PM -04:00 rac2

SQL> spool off
Thu Jun 12 17:02:05 EDT 2003 Starting up interface ...
Thu Jun 12 17:02:05 EDT 2003 en0 changed ...
Thu Jun 12 17:02:06 EDT 2003 Test finished ...
```

As the resulting output from the test script indicates, the running query is not completed in case the client network interface is brought down. The session, however, is reconnected to the next instance, rac2, and continues.

Active session - preconnect failover method

Using the preconnect failover method and running the test query while the interface is brought down on the node processing the query, the same result is observed as for the basic failover method (Example 6-25).

Example 6-25 Failing interface with active session (failover method=preconnect) - test script output

```
Thu Jun 12 17:14:57 EDT 2003 Test will connect to rac1p ...
Thu Jun 12 17:14:57 EDT 2003 Starting test ...
Thu Jun 12 17:14:57 EDT 2003 Starting query ...
Thu Jun 12 17:14:57 EDT 2003 Waiting for 30 seconds ...
Thu Jun 12 17:15:27 EDT 2003 Shutting down interface ...
Thu Jun 12 17:15:27 EDT 2003 en0 changed ...
Thu Jun 12 17:15:28 EDT 2003 Waiting for query to finish ...
```

```
Thu Jun 12 17:28:16 EDT 2003 Query finished ...
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
12-JUN-03 05.14.58.246059 PM -04:00 rac1

SQL> select count(*) from
  2  ( select * from dba_source
  3    union
  4    select * from dba_source
  5    union
  6    select * from dba_source
  7    union
  8    select * from dba_source
  9    union
 10    select * from dba_source);
select count(*) from
*
ERROR at line 1:
ORA-03113: end-of-file on communication channel


SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
12-JUN-03 05.28.16.097359 PM -04:00 rac2

SQL> spool off
Thu Jun 12 17:28:16 EDT 2003 Starting up interface ...
Thu Jun 12 17:28:16 EDT 2003 en0 changed ...
Thu Jun 12 17:28:16 EDT 2003 Test finished ...
```

These results were obtained with the parameter settings specified in this book. We could have found other settings that would have avoided the errors that occurred, and would have produced the correct query results, but, as we mentined in the beginning of this chapter, there are several ways to achieve the same objective, and we used just one of them.

6.2.5 Complete node fails

In the final test we bring the whole node down while the instance runs the test query. The failure and repair codes shown in Example 6-26 are included in the test script.

Example 6-26 Test script for node failure

```
# --- failing full node test
# --- code to be included in test script
#
# --- failure code:
#

echo "$(date) Failing node ..."
rsh node1 -n "touch trigger"

#
# --- repair code:
```

```
#

echo "$(date) Attention: node halted ..."
```

Instead of directly failing the node (using the **rsh shutdown** command), we choose to place a trigger file on the node that is to be brought down. On this node we set up a small loop that waits for the trigger file to appear before it halts the node (Example 6-27).

Example 6-27 Loop running on node to be failed

```
#!/bin/ksh

while sleep 2
do
   if [ -f /trigger ]
   then
      rm -f /trigger
      $1
      break
   else
      print ".\c"
   fi
done
```

The script takes a string as an argument and executes the string as a **ksh** command as soon as a file called *trigger* is found in the root directory. If the script is called wait_for_trigger and we issue the command **wait_for_trigger "halt -q"**, the node is stopped as soon as the file /trigger appears on the node.

In order to shorten the failover time, we set the network option tcp_keepidle to a value of 240, which corresponds to a timeout period of 2 minutes (the default value is 14400, which corresponds to 2 hours.)

Active session - basic failover method

Running the test query using the procedure described above and connecting using the basic failover method, the result obtained is shown in Example 6-28.

Example 6-28 Node failure with active session (failover method=basic) - test output

```
Thu Jun 12 16:17:50 EDT 2003 Test will connect to rac1f ...
Thu Jun 12 16:17:50 EDT 2003 Starting test ...
Thu Jun 12 16:17:50 EDT 2003 Starting query ...
Thu Jun 12 16:17:50 EDT 2003 Waiting for 30 seconds ...
Thu Jun 12 16:18:19 EDT 2003 Failing node ...
Thu Jun 12 16:18:19 EDT 2003 Waiting for query to finish ...
Thu Jun 12 16:34:12 EDT 2003 Query finished ...
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
12-JUN-03 04.17.49.933615 PM -04:00 rac1

SQL> select count(*) from
  2  ( select * from dba_source
  3    union
```

```
  4    select * from dba_source
  5    union
  6    select * from dba_source
  7    union
  8    select * from dba_source
  9    union
 10    select * from dba_source);

  COUNT(*)
----------
    119662

SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
12-JUN-03 04.34.12.427132 PM -04:00 rac2

SQL> spool off
Thu Jun 12 16:34:12 EDT 2003 Attention: node halted ...
Thu Jun 12 16:34:12 EDT 2003 Test finished ...
```

This test also produces the correct results, as expected. The query finishes without errors and the session is continued on the next available instance.

Active session - preconnect failover method

Repeating the previous test using a connection with the preconnect failover method enabled, we get the output shown in Example 6-29.

Example 6-29 Node failure with active session (failover method=preconnect) - test output

```
Fri Jun 13 10:29:01 EDT 2003 Test will connect to rac1p ...
Fri Jun 13 10:29:01 EDT 2003 Starting test ...
Fri Jun 13 10:29:01 EDT 2003 Starting query ...
Fri Jun 13 10:29:01 EDT 2003 Waiting for 30 seconds ...
Fri Jun 13 10:29:31 EDT 2003 Failing node ...
Fri Jun 13 10:29:31 EDT 2003 Waiting for query to finish ...
Fri Jun 13 10:44:43 EDT 2003 Query finished ...
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
13-JUN-03 10.29.02.174574 AM -04:00 rac1

SQL> select count(*) from
  2  ( select * from dba_source
  3    union
  4    select * from dba_source
  5    union
  6    select * from dba_source
  7    union
  8    select * from dba_source
  9    union
 10    select * from dba_source);

  COUNT(*)
----------
    119662
```

```
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
13-JUN-03 10.44.42.544695 AM -04:00 rac2

SQL> spool off
Fri Jun 13 10:44:43 EDT 2003 Attention: node halted ...
Fri Jun 13 10:44:43 EDT 2003 Test finished ...
```

Using the preconnect failover method, we also obtain the correct result for the query and the session is continued on the backup instance.

6.3 Platform availability tests

This section describes some tests that demonstrate how Oracle9*i* RAC reacts in case a server component fails, which could potentially disturb the instance processing the query. We consider the following test cases:

- GPFS subsystem failure on one node
- GPFS network interface failure on one node
- Nodes failure

As for the database availability tests, we observe the behavior of a session connected to the instance on the node where the failure occurs, and monitor the result of the query, as well as the resulting reconnect.

6.3.1 GPFS subsystem failure on one node

For this test we stop the GPFS subsystem on one node where a query is running. We use the same table, named DATA, which contains 5.2 million rows, and perform a non-parallel table scan that reads data blocks from the datafiles located on a GPFS file system.

Stopping the GPFS subsystem leads to the loss of both the data files and the Oracle9*i* RAC executables, since they are also located on a GPFS file system.

We adapt the test script by including the failure and repair codes shown in Example 6-30.

Example 6-30 Failing GPFS subsystem - failure and repair codes for test script

```
# --- failing GPFS subsystem interface test
# --- code to be included in test script
#
# --- failure code:
#

echo "$(date) Shutting down GPFS ..."
rsh node1 -n "/usr/lpp/mmfs/bin/mmshutdown >/dev/null 2>&1"

#
# --- repair code:
#
```

```
echo "$(date) Starting up GPFS ..."
rsh node1 -n "/usr/lpp/mmfs/bin/mmstartup >/dev/null 2>&1"
echo "$(date) Attention: please clean up processes and ipc ..."
echo "$(date) Listener NOT started automatically ..."
echo "$(date) Instance NOT started automatically ..."
```

Running the adapted test script results in the output shown in Example 6-31.

Example 6-31 GPFS fails on one node - test script output

```
Mon Jun  9 17:41:50 EDT 2003 Test will connect to rac1f ...
Mon Jun  9 17:41:50 EDT 2003 Starting test ...
Mon Jun  9 17:41:50 EDT 2003 Starting query ...
Mon Jun  9 17:41:50 EDT 2003 Waiting for 30 seconds ...
Mon Jun  9 17:42:20 EDT 2003 Shutting down GPFS ...
Mon Jun  9 17:43:27 EDT 2003 Waiting for query to finish ...
Mon Jun  9 17:56:07 EDT 2003 Query finished ...
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
09-JUN-03 05.41.50.552578 PM -04:00 rac1

SQL> select /*+ noparallel */ count(*) from data;

  COUNT(*)
----------
   5200000

SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
09-JUN-03 05.56.06.986343 PM -04:00 rac2

SQL> spool off
Mon Jun  9 17:56:07 EDT 2003 Starting up GPFS ...
Mon Jun  9 17:56:11 EDT 2003 Attention: please clean up processes and ipc ...
Mon Jun  9 17:56:11 EDT 2003 Listener NOT started automatically ...
Mon Jun  9 17:56:11 EDT 2003 Instance NOT started automatically ...
Mon Jun  9 17:56:11 EDT 2003 Test finished ...
```

As we can see from the previous example listing, the query is completed successfully, and the session is reconnected to the next available instance.

After this test, the instance on node1 has crashed and also the listener is not running (missing code, due to missing file system).

Before we can restart the instance, we have to clean up the shared memory segments left by the old instance on the node. We list the shared memory segments using:

```
{node1:root}/-> ipcs -m|grep oracle
IPC status from /dev/mem as of Tue Jun 04 13:40:33 EDT 2003
T        ID     KEY        MODE       OWNER    GROUP
Shared Memory:
m        17 0x6756c9e8 --rw-r-----   oracle      dba
{node1:root}/->
```

and note the ID of the shared memory segment belonging to user oracle. We delete this segment with:

```
ipcrm -m ID
```

where `ID` is the ID of the shared memory segment identified in the previous step. In case there is more than one segment left from the old instance, this has to be repeated for all the segments.

Important: You must *not* delete shared memory segments belonging to other instances.

6.3.2 GPFS network interface failure on one node

For this test we use the same query (table scan) as for the previous test. This time we do not shut down the whole GPFS on one node, but just the network interface that connects that node to the GPFS network.

The failure and repair codes to be included into the test script are shown in Example 6-32.

Example 6-32 Failing GPFS network interface - failure and repair codes for test script

```
# --- failing GPFS network interface test
# --- code to be included in test script
#
# --- failure code:
#

echo “$(date) Shutting down GPFS interface ...”
rsh node1 -n “echo \”$(date) $(chdev -l en3 -a state=down) ...\” ”

#
# --- repair code:
#

echo “$(date) Starting up GPFS interface ...”
rsh node1 -n “echo \”$(date) $(chdev -l en3 -a state=up) ...\” ”
echo “$(date) Attention: please clean up processes and ipc ...”
echo “$(date) GPFS NOT started automatically ...”
echo “$(date) Listener NOT started automatically ...”
echo “$(date) Instance NOT started automatically ...”
```

GPFS network interface failure makes the Oracle9*i* RAC datafiles and executables unavailable (no access to the GPFS file system).

Running the adapted test script produces the output shown in Example 6-33.

Example 6-33 GPFS network failure - test results

```
Mon Jun  9 18:52:47 EDT 2003 Test will connect to rac1f ...
Mon Jun  9 18:52:47 EDT 2003 Starting test ...
Mon Jun  9 18:52:47 EDT 2003 Starting query ...
Mon Jun  9 18:52:47 EDT 2003 Waiting for 30 seconds ...
Mon Jun  9 18:53:17 EDT 2003 Shutting down GPFS interface ...
Mon Jun  9 18:53:17 EDT 2003 en3 changed ...
Mon Jun  9 18:53:17 EDT 2003 Waiting for query to finish ...
Mon Jun  9 18:56:24 EDT 2003 Query finished ...
```

```
SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
09-JUN-03 06.52.49.787107 PM -04:00 rac1

SQL> select /*+ noparallel */ count(*) from data;

  COUNT(*)
----------
   5200000

SQL> @time_instance

CURRENT_TIMESTAMP                   INSTANCE_NAME
----------------------------------- ----------------
09-JUN-03 06.56.24.361454 PM -04:00 rac2

SQL> spool off
Mon Jun  9 18:56:24 EDT 2003 Starting up GPFS interface ...
Mon Jun  9 18:56:24 EDT 2003 en3 changed ...
Mon Jun  9 18:56:24 EDT 2003 Attention: please clean up processes and ipc ...
Mon Jun  9 18:56:24 EDT 2003 GPFS NOT started automatically ...
Mon Jun  9 18:56:24 EDT 2003 Listener NOT started automatically ...
Mon Jun  9 18:56:24 EDT 2003 Instance NOT started automatically ...
Mon Jun  9 18:56:24 EDT 2003 Test finished ...
```

As in the previous test, the result is correct and as expected. Again, Oracle9*i* crashes when it loses its executables, and the shared memory segments have to be cleared before the instance can be restarted. The GPFS file systems, however, are still mounted and are available again, as soon as the network interface is brought up.

6.4 Summary of tests

In the previous sections, 6.2, “Database availability tests” on page 174 and 6.3, “Platform availability tests” on page 191, we investigate the behavior of three different types of query, an operating system **sleep** for an idle sqlplus session, and two versions of **select** statements. One of them contains a multiple union of the dba_source view, the other is a full table scan of a table DATA with 5.2 million rows producing some load on the GPFS subsystem.

We do not present all possible combinations, but pick those we think are the most meaningful for functionality and component utilization.

Table 6-2 summarizes the results of our tests.

Table 6-2 Summary of high availability tests

Failing Component	Type of test query			Session did fail over?
	!sleep 120	select (... union...);	select ... from data;	
Listener	-[a]	OK	-	Not required
Instance	OK	OK	-	Yes
Interconnect	-	-	OK	Not required

Failing Component	Type of test query			Session did fail over?
	!sleep 120	select (... union...);	select ... from data;	
Interface	OK	Error	-	Yes
GPFS	-	-	OK	Yes
GPFS interface	-	-	OK	Yes
Node	-	OK	-	Yes

a. Table entries containing "-" represent tests that are not reported in this book.

In general (with the exception of the failed client network interface, see 6.2.4, "Client network interface fails on one node" on page 185) you can see that the failover functions are working as expected, once the proper configuration files and system parameters are in place.

A

Operating system fileset levels

This appendix provides a complete list of the fileset levels installed on our platform.

The following software was installed on our test environment:

- AIX 5.2 Maintenance Level 1
- RSCT 2.3.1
- GPFS 2.1
- HACMP 4.5
- Enterprise Storage Server 2105 32.6.100.13
- Enterprise Storage Server SDD 5.1.0
- Fibre Channel drivers
- Java 1.3.1

AIX 5.2 ML1 base operating system filesets

```
{node1:root}/home/michel-> lslpp -lOu bos.*
  Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  bos.64bit                 5.2.0.10  COMMITTED  Base Operating System 64 bit
                                                 Runtime
  bos.acct                  5.2.0.10  COMMITTED  Accounting Services
  bos.adt.base              5.2.0.10  COMMITTED  Base Application Development
                                                 Toolkit
  bos.adt.debug             5.2.0.10  COMMITTED  Base Application Development
                                                 Debuggers
  bos.adt.graphics           5.2.0.0  COMMITTED  Base Application Development
                                                 Graphics Include Files
  bos.adt.include           5.2.0.10  COMMITTED  Base Application Development
                                                 Include Files
  bos.adt.lib               5.2.0.10  COMMITTED  Base Application Development
                                                 Libraries
  bos.adt.libm              5.2.0.10  COMMITTED  Base Application Development
                                                 Math Library
  bos.adt.prof              5.2.0.10  COMMITTED  Base Profiling Support
  bos.adt.prt_tools          5.2.0.0  COMMITTED  Printer Support Development
                                                 Toolkit
  bos.adt.samples           5.2.0.10  COMMITTED  Base Operating System Samples
  bos.adt.sccs               5.2.0.0  COMMITTED  SCCS Application Development
                                                 Toolkit
  bos.adt.syscalls          5.2.0.10  COMMITTED  System Calls Application
                                                 Development Toolkit
  bos.adt.utils              5.2.0.0  COMMITTED  Base Application Development
                                                 Utilities - lex and yacc
  bos.alt_disk_install.boot_images
                            5.2.0.10  COMMITTED  Alternate Disk Installation
                                                 Disk Boot Images
  bos.alt_disk_install.rte  5.2.0.10  COMMITTED  Alternate Disk Installation
                                                 Runtime
  bos.cdmount               5.2.0.10  COMMITTED  CD/DVD Automount Facility
  bos.clvm.enh              5.2.0.10  COMMITTED  Enhanced Concurrent Logical
                                                 Volume Manager
  bos.content_list           5.2.0.0  COMMITTED  AIX Release Content List
  bos.diag.com              5.2.0.10  COMMITTED  Common Hardware Diagnostics
  bos.diag.rte              5.2.0.10  COMMITTED  Hardware Diagnostics
  bos.diag.util             5.2.0.10  COMMITTED  Hardware Diagnostics Utilities
  bos.docregister.com        5.2.0.0  COMMITTED  Docregister Common
  bos.docsearch.client.Dt    5.2.0.0  COMMITTED  DocSearch Client CDE
                                                 Application Integration
  bos.docsearch.client.com   5.2.0.0  COMMITTED  DocSearch Client Common Files
  bos.docsearch.rte          5.2.0.0  COMMITTED  DocSearch Runtime
  bos.help.msg.en_US.com     5.2.0.0  COMMITTED  WebSM/SMIT Context Helps -
                                                 U.S. English
  bos.help.msg.en_US.smit    5.2.0.0  COMMITTED  SMIT Context Helps - U.S.
                                                 English
  bos.html.en_US.topnav.navigate
                             5.2.0.0  COMMITTED  Top Level Navigation - U.S.
                                                 English
  bos.iconv.com             5.2.0.10  COMMITTED  Common Language to Language
                                                 Converters
  bos.iconv.ucs.com         5.2.0.10  COMMITTED  Unicode Base Converters for
                                                 AIX Code Sets/Fonts
  bos.loc.iso.en_US          5.2.0.0  COMMITTED  Base System Locale ISO Code
```

```
                                                 Set - U.S. English
bos.man.en_US.cmds         5.2.0.0  COMMITTED  AIX Man Commands - U.S.
                                                 English
bos.mh                     5.2.0.0  COMMITTED  Mail Handler
bos.mp                    5.2.0.10  COMMITTED  Base Operating System
                                                 Multiprocessor Runtime
bos.mp64                  5.2.0.10  COMMITTED  Base Operating System 64-bit
                                                 Multiprocessor Runtime
bos.msg.en_US.alt_disk_install.rte
                           5.2.0.0  COMMITTED  Alternate Disk Install Msgs -
                                                 U.S. English
bos.msg.en_US.diag.rte     5.2.0.0  COMMITTED  Hardware Diagnostics Messages
                                                 - U.S. English
bos.msg.en_US.docregister.com
                           5.2.0.0  COMMITTED  Docregister Common Messages -
                                                 U.S. English
bos.msg.en_US.docsearch.client.Dt
                           5.2.0.0  COMMITTED  DocSearch CDE Action - U.S.
                                                 English
bos.msg.en_US.docsearch.client.com
                           5.2.0.0  COMMITTED  DocSearch Common Messages -
                                                 U.S. English
bos.msg.en_US.mp           5.2.0.0  COMMITTED  Base Operating System MP Msgs
                                                 - U.S. English
bos.msg.en_US.net.tcp.client
                           5.2.0.0  COMMITTED  TCP/IP Messages - U.S. English
bos.msg.en_US.rte          5.2.0.0  COMMITTED  Base OS Runtime Messages -
                                                 U.S. English
bos.msg.en_US.txt.tfs      5.2.0.0  COMMITTED  Text Formatting Services Msgs
                                                 - U.S. English
bos.net.ncs                5.2.0.0  COMMITTED  Network Computing System 1.5.1
bos.net.nfs.client        5.2.0.10  COMMITTED  Network File System Client
bos.net.nfs.server        5.2.0.10  COMMITTED  Network File System Server
bos.net.nis.client        5.2.0.10  COMMITTED  Network Information Service
                                                 Client
bos.net.nis.server        5.2.0.10  COMMITTED  Network Information Service
                                                 Server
bos.net.snapp              5.2.0.0  COMMITTED  System Networking Analysis and
                                                 Performance Pilot
bos.net.tcp.adt           5.2.0.10  COMMITTED  TCP/IP Application Toolkit
bos.net.tcp.client        5.2.0.10  COMMITTED  TCP/IP Client Support
bos.net.tcp.server        5.2.0.10  COMMITTED  TCP/IP Server
bos.net.tcp.smit          5.2.0.10  COMMITTED  TCP/IP SMIT Support
bos.net.uucp              5.2.0.10  COMMITTED  Unix to Unix Copy Program
bos.perf.diag_tool        5.2.0.10  COMMITTED  Performance Diagnostic Tool
bos.perf.libperfstat      5.2.0.10  COMMITTED  Performance Statistics Library
                                                 Interface
bos.perf.perfstat         5.2.0.10  COMMITTED  Performance Statistics
                                                 Interface
bos.perf.proctools        5.2.0.10  COMMITTED  Proc Filesystem Tools
bos.perf.tools            5.2.0.10  COMMITTED  Base Performance Tools
bos.perf.tune             5.2.0.10  COMMITTED  Performance Tuning Support
bos.rte                   5.2.0.10  APPLIED    Base Operating System Runtime
bos.rte.Dt                 5.2.0.0  COMMITTED  Desktop Integrator
bos.rte.ILS               5.2.0.10  APPLIED    International Language Support
bos.rte.SRC               5.2.0.10  APPLIED    System Resource Controller
bos.rte.X11                5.2.0.0  COMMITTED  AIXwindows Device Support
bos.rte.aio               5.2.0.10  APPLIED    Asynchronous I/O Extension
bos.rte.archive           5.2.0.10  APPLIED    Archive Commands
bos.rte.bind_cmds         5.2.0.10  APPLIED    Binder and Loader Commands
```

```
bos.rte.boot             5.2.0.10  APPLIED    Boot Commands
bos.rte.bosinst          5.2.0.10  APPLIED    Base OS Install Commands
bos.rte.commands         5.2.0.10  APPLIED    Commands
bos.rte.compare           5.2.0.0  COMMITTED  File Compare Commands
bos.rte.console          5.2.0.10  APPLIED    Console
bos.rte.control          5.2.0.10  APPLIED    System Control Commands
bos.rte.cron             5.2.0.10  APPLIED    Batch Operations
bos.rte.date             5.2.0.10  APPLIED    Date Control Commands
bos.rte.devices          5.2.0.10  APPLIED    Base Device Drivers
bos.rte.devices_msg      5.2.0.10  APPLIED    Device Driver Messages
bos.rte.diag             5.2.0.10  APPLIED    Diagnostics
bos.rte.edit             5.2.0.10  APPLIED    Editors
bos.rte.filesystem       5.2.0.10  APPLIED    Filesystem Administration
bos.rte.iconv            5.2.0.10  APPLIED    Language Converters
bos.rte.ifor_ls          5.2.0.10  APPLIED    iFOR/LS Libraries
bos.rte.im               5.2.0.10  APPLIED    Input Methods
bos.rte.install          5.2.0.10  APPLIED    LPP Install Commands
bos.rte.jfscomp           5.2.0.0  COMMITTED  JFS Compression
bos.rte.libc             5.2.0.10  APPLIED    libc Library
bos.rte.libcfg            5.2.0.0  COMMITTED  libcfg Library
bos.rte.libcur           5.2.0.10  APPLIED    libcurses Library
bos.rte.libdbm            5.2.0.0  COMMITTED  libdbm Library
bos.rte.libnetsvc         5.2.0.0  COMMITTED  Network Services Libraries
bos.rte.libpthreads      5.2.0.10  APPLIED    libpthreads Library
bos.rte.libqb             5.2.0.0  COMMITTED  libqb Library
bos.rte.libs             5.2.0.10  APPLIED    libs Library
bos.rte.loc              5.2.0.10  APPLIED    Base Locale Support
bos.rte.lvm              5.2.0.10  APPLIED    Logical Volume Manager
bos.rte.man              5.2.0.10  APPLIED    Man Commands
bos.rte.methods          5.2.0.10  APPLIED    Device Config Methods
bos.rte.misc_cmds        5.2.0.10  APPLIED    Miscellaneous Commands
bos.rte.net               5.2.0.0  COMMITTED  Network
bos.rte.odm              5.2.0.10  APPLIED    Object Data Manager
bos.rte.printers         5.2.0.10  APPLIED    Front End Printer Support
bos.rte.security         5.2.0.10  APPLIED    Base Security Function
bos.rte.serv_aid         5.2.0.10  APPLIED    Error Log Service Aids
bos.rte.shell            5.2.0.10  APPLIED    Shells (bsh, ksh, csh)
bos.rte.streams          5.2.0.10  APPLIED    Streams Libraries
bos.rte.tty              5.2.0.10  APPLIED    Base TTY Support and Commands
bos.sysmgt.loginlic       5.2.0.0  COMMITTED  License Management
bos.sysmgt.nim.client    5.2.0.10  COMMITTED  Network Install Manager -
                                              Client Tools
bos.sysmgt.quota          5.2.0.0  COMMITTED  Filesystem Quota Commands
bos.sysmgt.serv_aid      5.2.0.10  COMMITTED  Software Error Logging and
                                              Dump Service Aids
bos.sysmgt.smit          5.2.0.10  COMMITTED  System Management Interface
                                              Tool (SMIT)
bos.sysmgt.sysbr         5.2.0.10  COMMITTED  System Backup and BOS Install
                                              Utilities
bos.sysmgt.trace         5.2.0.10  COMMITTED  Software Trace Service Aids
bos.terminfo.rte          5.2.0.0  COMMITTED  Run-time Environment for AIX
                                              Terminals
bos.txt.spell             5.2.0.0  COMMITTED  Writer's Tools Commands
bos.txt.tfs               5.2.0.0  COMMITTED  Text Formatting Services
                                              Commands
bos.up                   5.2.0.10  COMMITTED  Base Operating System
                                              Uniprocessor Runtime
```

RSCT 2.3.1 filesets

```
{node4:root}/home/michel-> lslpp -lOu rsct*
  Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  rsct.basic.hacmp           2.3.1.0  COMMITTED  RSCT Basic Function (HACMP/ES
                                                 Support)
  rsct.basic.rte             2.3.1.0  COMMITTED  RSCT Basic Function
  rsct.basic.sp              2.3.1.0  COMMITTED  RSCT Basic Function (PSSP
                                                 Support)
  rsct.clients.rte     99.99.999.999  COMMITTED  Supersede Entry - Not really
                                                 installed
  rsct.compat.basic.hacmp    2.3.1.0  COMMITTED  RSCT Event Management Basic
                                                 Function (HACMP/ES Support)
  rsct.compat.basic.rte      2.3.1.0  COMMITTED  RSCT Event Management Basic
                                                 Function
  rsct.compat.basic.sp       2.3.1.0  COMMITTED  RSCT Event Management Basic
                                                 Function (PSSP Support)
  rsct.compat.clients.hacmp  2.3.1.0  COMMITTED  RSCT Event Management Client
                                                 Function (HACMP/ES Support)
  rsct.compat.clients.rte    2.3.1.0  COMMITTED  RSCT Event Management Client
                                                 Function
  rsct.compat.clients.sp     2.3.1.0  COMMITTED  RSCT Event Management Client
                                                 Function (PSSP Support)
  rsct.core.auditrm          2.3.1.0  COMMITTED  RSCT Audit Log Resource
                                                 Manager
  rsct.core.errm             2.3.1.0  COMMITTED  RSCT Event Response Resource
                                                 Manager
  rsct.core.fsrm             2.3.1.0  COMMITTED  RSCT File System Resource
                                                 Manager
  rsct.core.gui              2.3.1.0  COMMITTED  RSCT Graphical User Interface
  rsct.core.hostrm           2.3.1.0  COMMITTED  RSCT Host Resource Manager
  rsct.core.rmc              2.3.1.0  COMMITTED  RSCT Resource Monitoring and
                                                 Control
  rsct.core.sec              2.3.1.0  COMMITTED  RSCT Security
  rsct.core.sr               2.3.1.0  COMMITTED  RSCT Registry
  rsct.core.utils            2.3.1.0  COMMITTED  RSCT Utilities
  rsct.msg.EN_US.core.auditrm
                             2.3.0.0  COMMITTED  RSCT Audit Log RM Msgs - U.S.
                                                 English (UTF)
  rsct.msg.EN_US.core.errm   2.3.0.0  COMMITTED  RSCT Event Response RM Msgs -
                                                 U.S. English (UTF)
  rsct.msg.EN_US.core.fsrm   2.3.0.0  COMMITTED  RSCT File System RM Msgs -
                                                 U.S. English (UTF)
  rsct.msg.EN_US.core.gui    2.3.0.0  COMMITTED  RSCT GUI Msgs - U.S. English
                                                 (UTF)
  rsct.msg.EN_US.core.hostrm
                             2.3.0.0  COMMITTED  RSCT Host RM Msgs - U.S.
                                                 English (UTF)
  rsct.msg.EN_US.core.rmc    2.3.0.0  COMMITTED  RSCT RMC Msgs - U.S. English
                                                 (UTF)
  rsct.msg.EN_US.core.sec    2.3.0.0  COMMITTED  RSCT Security Msgs - U.S.
                                                 English (UTF)
  rsct.msg.EN_US.core.sr     2.3.0.0  COMMITTED  RSCT Registry Msgs - U.S.
                                                 English (UTF)
  rsct.msg.EN_US.core.utils  2.3.0.0  COMMITTED  RSCT Utilities Msgs - U.S.
                                                 English (UTF)
  rsct.msg.en_US.basic.rte   2.3.0.0  COMMITTED  RSCT Basic Msgs - U.S. English
  rsct.msg.en_US.core.auditrm
```

```
                            2.3.0.0  COMMITTED  RSCT Audit Log RM Msgs - U.S.
                                                English
  rsct.msg.en_US.core.errm  2.3.0.0  COMMITTED  RSCT Event Response RM Msgs -
                                                U.S. English
  rsct.msg.en_US.core.fsrm  2.3.0.0  COMMITTED  RSCT File System RM Msgs -
                                                U.S. English
  rsct.msg.en_US.core.gui   2.3.0.0  COMMITTED  RSCT GUI Msgs - U.S. English
  rsct.msg.en_US.core.gui.com
                            2.3.0.0  COMMITTED  RSCT GUI JAVA Msgs - U.S.
                                                English
  rsct.msg.en_US.core.hostrm
                            2.3.0.0  COMMITTED  RSCT Host RM Msgs - U.S.
                                                English
  rsct.msg.en_US.core.rmc   2.3.0.0  COMMITTED  RSCT RMC Msgs - U.S. English
  rsct.msg.en_US.core.rmc.com
                            2.3.0.0  COMMITTED  RSCT RMC JAVA Msgs - U.S.
                                                English
  rsct.msg.en_US.core.sec   2.3.0.0  COMMITTED  RSCT Security Msgs - U.S.
                                                English
  rsct.msg.en_US.core.sr    2.3.0.0  COMMITTED  RSCT Registry Msgs - U.S.
                                                English
  rsct.msg.en_US.core.utils 2.3.0.0  COMMITTED  RSCT Utilities Msgs - U.S.
                                                English
```

GPFS 2.1 filesets

```
{node4:root}/home/michel-> lslpp -lOu mmfs*
  Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  mmfs.base.cmds             3.5.0.6  APPLIED    GPFS File Manager Commands
  mmfs.base.rte              3.5.0.6  APPLIED    GPFS File Manager
  mmfs.gpfs.rte              2.1.0.6  APPLIED    GPFS File Manager
  mmfs.msg.en_US             3.5.0.4  APPLIED    GPFS Server Messages - U.S.
                                                 English
```

These are the contents of the file /var/mmfs/etc/mmfs_vartonvg.scr.

Only one version of this script is working properly. The varyonvg and mount could fail. If you experience this behavior, a workaround is to manually **varyonvg gpfs0vg** with the **-u** option, and then mount your file system.

When running the command **mmfs_startup -a**, if you get the message `6027-419 Failed to read a file system descriptor`, then check whether the bold lines in the following code appear in your script—and then adjust your script accordingly.

```
#!/bin/ksh
# @(#)60 1.24 src/avs/fs/mmfs/ts/phoenix/mmfs_varyonvg.scr, mmfs, avs_rwyn 10/16/02
23:38:17
#############################################################
#                                                           #
#  Usage:                                                   #
#    mmfs_varyonvg.scr [setfenceid] [diskName]              #
#                                                           #
#  This shell script is called with:                        #
#    2 arguments from devOpen() function if device open fails.    #
#                                                           #
#   If $1  = "1" verify fence id.                           #
```

```
#   Do not varyon the disk if the fence id is not set correctly.  #
#   If diskName is passed, find the stripe group and activate     #
#   all the disks that belong to that file system.                #
#   Otherwise, activate all the disks found in the mmsdrfs file.  #
#                                                                 #
###################################################################

# Include global declarations and service routines
. /usr/lpp/mmfs/bin/mmglobfuncs
. /usr/lpp/mmfs/bin/mmsdrfsdef
. /usr/lpp/mmfs/bin/mmfsfuncs

[[ -n $DEBUG || -n $DEBUGmmfs_varyonvg ]] && set -x
$mmTRACE_ENTER "$*"

# This function is not needed in Linux.
[[ $osName = Linux ]] && exit 0

diskName=""
fsName=""
integer found=0


#####################################################################
#                                                                   #
# This function activates a volume group when fencing is in use.    #
#                                                                   #
#####################################################################
function verifyAndActivateVg
{
  typeset vgname=$1

  # Verify whether the node is fenced out from the disk or not.
  # If the node is fenced out, do not activate.
  $mmisnodefenced -v $vgname $ourNodeNumber
  if [[ $? -eq 0 ]]
  then
    # Activate the disk.
    $varyonvg -u $vgname
    if [[ $? = 0 ]]
    then
# bch - fix msg
      print -u2 "$(date): Activated $vgname successfully."
    else
# bch - fix msg
      print -u2 "$(date): Failed activating $vgname."
    fi
  fi

}    # ---- end of function verifyAndActivateVg ----------


#########################################################################
#                                                                       #
# This function activates a volume group when disk leasing is in use. #
#                                                                       #
#########################################################################
function activateVg
{
  typeset vgname=$1
```

```
  # Activate the disk.
  $varyonvg -u $vgname
  if [[ $? = 0 ]]
  then
# bch - fix msg
    print -u2 "$(date): Activated $vgname successfully."
  else
# bch - fix msg
    print -u2 "$(date): Failed activating $vgname."
  fi

}    # ---- end of function activateVg ----------


#########################################################################
#                                                                       #
# This function checks whether a disk is an SSA disk.                   #
# It accepts one argument, the logical volume name of the disk.         #
#                                                                       #
# It returns 1 if the input disk is an SSA disk, 0 otherwise.           #
#                                                                       #
#########################################################################
function is_ssadisk
{
  typeset lvname=$1

  # Check whether the disk is an SSA disk.
  hdisk=$($mmfencevalidparms -l $lvname)
  LANG=C $lsdev -Ccdisk -l $hdisk | $grep SSA > /dev/null
  if [[ $? -eq 0 ]]
  then
    retval=1
  else
    retval=0
  fi

  return $retval

}    # ---- end of function is_ssadisk ----------


#########################################################################
#
# Main function
#
#########################################################################

# Default argument is 0
[[ $argc < 1 ]] && set -- 0

# Check the number of parms that were passed.
if [[ $argc > 1 ]]
then
  # verifyFenceid and diskName were passed.
  verifyFenceid=$arg1
  diskName=$arg2
fi

# Get the node number for this node.
```

```
[[ -z $MMMODE || -z $primaryServer ]] && determineMode
getLocalNodeData

# Get a list containing the physical disk name, pvid, and vgname
# for all known volume groups on this node.
LANG=C $lspv > $volGroupFile

if [[ -n $diskName ]]
then
  fsName=$($mmcommon getSGDevice $diskName)
fi

# Create a temp file containing the disks (along with their subtypes
# and pvids) to be varied on.
getDiskData $tmpfile $fsName

# Obtain the disk leasing flag value.
useDiskLease=$($tsctl showCfgValue useDiskLease 2>/dev/null)
rc=$?
[[ $useDiskLease = 1 ]] && useDiskLease=yes

# If the useDiskLease value is still unknown, get it from the mmsdrfs file.
if [[ $rc -ne 0 || -z $useDiskLease ]]
then
  useDiskLease=$($awk '                              \
    BEGIN { diskLease = "no" }                       \
    { if ($1 == "useDiskLease") diskLease = $2 }  \
    END { print diskLease }                          \
  ' $mmfscfgFile 2>/dev/null)
fi
[[ $useDiskLease != yes ]] && useDiskLease=no

checkSSAFenceId=yes
goodSSAenv=true

exec 3<&-
exec 3< $tmpfile
while read -u3 diskLine
do
  # Extract the lvname, disk subtype, and pvid for each disk.
  set -- $diskLine
  lvname=$1
  disksubtype=$2
  pvid=$3

  # Create the vgname starting from the lvname.
  # bch - The following LOC requires that our naming convention be in effect,
  # bch - effectively precluding lvs manually created w/o the use of mmcrlv.
  # bch - Is this a compatibility problem for old releases?
  vgname=${lvname%lv}vg   # replace the ending "lv" with "vg" to get vgname

  # If the disk is not known on this node, try to import it.
  lspvLine=$($grep $vgname $volGroupFile)
  if [[ $? = 1 ]]
  then
    importDisk $lvname $vgname $pvid $volGroupFile
    # If we couldn't make the disk known on this node, skip to the next disk.
    if [[ $? = 1 ]]
    then
# bch - fix msg
```

```
      print -u2 "Skipping disk $lvname on node $ourNodeName."
      continue
    fi
    lspvLine=$($grep $vgname $volGroupFile)
  fi

  # Use the lslv command to check whether the volume group is active or not.
  active=$(LANG=C $lslv -L $lvname | grep "VG STATE" | awk '{print $3}' | awk -F/ '{print
$1}')

  [[ $diskName = $lvname ]] && found=1

  # Check the fencing setup if required.
  if [[ $disksubtype = "pr" && $useDiskLease = no ]]
  then
    # Check the PR fencing setup since this is a PR disk.
    hdisk=$(echo $lspvLine | $awk '{print $1}')
    checkPRfencing $hdisk
    if [[ $? = 1 ]]
    then
      printErrorMsg 456 $mmcmd
      rc=1
    else
      # Call routine to check fencing and activate the vg if needed.
      [[ $active != "active" ]] && verifyAndActivateVg $vgname
    fi
  elif [[ $disksubtype = ssa && $checkSSAFenceId = yes && $useDiskLease = no ]]
  then
    # Check the SSA fencing setup for the first-encountered SSA disk.
    checkSSAfencing
    if [[ $? = 1 ]]
    then
      goodSSAenv=false
      rc=1
    else
      # Unless this is an SSA disk and an SSA environment problem was found,
      # call routine to check fencing and activate the vg if needed.
      if [[ $goodSSAenv = true ]]
      then
        [[ $active != "active" ]] && verifyAndActivateVg $vgname
      fi
    fi
    checkSSAFenceId=no
  else
    # Here if disk leasing is being used.  Activate the vg if needed.
    [[ $active != "active" ]] && activateVg $vgname
  fi
done

# If a disk was passed but it was not found in the mmsdrfs file,
# then try to activate that disk also.  We will not be able to handle
# the case where the disk has not been imported yet, since we have no way
# to obtain the pvid of the disk, if it is not listed in the mmsdrfs file.
if [[ $argc = 2 && $found = 0 && $verifyFenceid = 1 ]]
then
  # Run lslv command to determine whether the volume group is active or not.
  lslvLine=$(LANG=C $lslv -L $diskName 2> /dev/null | grep "VG STATE")
  if [[ $? = 0 ]]
  then
    active=$(echo $lslvLine | awk '{print $3}' | awk -F/ '{print $1}')
```

```
    vgname=$(LANG=C $lslv -L $diskName | grep "VOLUME GROUP" | awk '{print $6}')
    if [[ $useDiskLease = no ]]
    then
      # Here if fencing is being used.
      # If this is an SSA disk and the SSA fence id has not been
      # verified yet, check whether the SSA fence id is set correctly.
      # We do not check for PR disks in this case, because we require
      # that PR disks must have been created by mmcrlv, and therefore
      # we can count on the disk being listed in the mmsdrfs file.
      is_ssadisk $diskName
      SSAdisk=$?
      if [[ $SSAdisk = 1 && $checkSSAFenceId = yes ]]
      then
        # Check that the SSA fencing setup is correct.
        checkSSAfencing
        if [[ $? = 1 ]]
        then
          goodSSAenv=false
          rc=1
        fi
      fi
      # Unless this is an SSA disk and an SSA environment problem was found,
      # call routine to check the fencing and activate the vg if needed.
      if [[ $SSAdisk = 0 || $goodSSAenv = true ]]
      then
        [[ $active != "active" ]] && verifyAndActivateVg $vgname
      fi
    else
      # Here if disk leasing is being used.  Activate the vg if needed.
      [[ $active != "active" ]] && activateVg $vgname
    fi  # if [[ $useDiskLease = no ]]
  else
# bch - fix msg
    print -u2 "Disk $diskName is not known on node $ourNodeName."
  fi
fi

# Always return zero.
cleanupAndExit 0
```

HACMP 4.5 filesets

```
{node4:root}/home/michel-> lslpp -lOu cluster*
Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  cluster.adt.es.client.demos
                             4.5.0.2  COMMITTED  ES Client Demos
  cluster.adt.es.client.include
                             4.5.0.4  COMMITTED  ES Client Include Files
  cluster.adt.es.client.samples.clinfo
                             4.5.0.5  COMMITTED  ES Client CLINFO Samples
  cluster.adt.es.client.samples.clstat
                             4.5.0.2  COMMITTED  ES Client Clstat Samples
  cluster.adt.es.client.samples.demos
                             4.5.0.3  COMMITTED  ES Client Demos Samples
  cluster.adt.es.client.samples.libcl
                             4.5.0.3  COMMITTED  ES Client LIBCL Samples
  cluster.adt.es.java.demo.monitor
```

```
                            4.5.0.2  COMMITTED  ES Web Based Monitor Demo
  cluster.adt.es.server.demos
                            4.5.0.2  COMMITTED  ES Server Demos
  cluster.adt.es.server.samples.demos
                            4.5.0.3  COMMITTED  ES Server Sample Demos
  cluster.adt.es.server.samples.images
                            4.5.0.2  COMMITTED  ES Server Sample Images
  cluster.doc.en_US.es.html  4.5.0.4  COMMITTED  HAES Web-based HTML
                                                 Documentation - U.S. English
  cluster.doc.en_US.es.pdf   4.5.0.4  COMMITTED  HAES PDF Documentation - U.S.
                                                 English
  cluster.doc.en_US.html     4.5.0.4  COMMITTED  HACMP Web-based HTML
                                                 Documentation - U.S. English
  cluster.doc.en_US.pdf      4.5.0.4  COMMITTED  HACMP PDF Documentation - U.S.
                                                 English
  cluster.es.client.lib      4.5.0.4  COMMITTED  ES Client Libraries
  cluster.es.client.rte      4.5.0.5  COMMITTED  ES Client Runtime
  cluster.es.client.utils    4.5.0.3  COMMITTED  ES Client Utilities
  cluster.es.clvm.rte        4.5.0.2  COMMITTED  ES for AIX Concurrent Access
  cluster.es.cspoc.cmds      4.5.0.5  COMMITTED  ES CSPOC Commands
  cluster.es.cspoc.dsh       4.5.0.2  COMMITTED  ES CSPOC dsh
  cluster.es.cspoc.rte       4.5.0.6  COMMITTED  ES CSPOC Runtime Commands
  cluster.es.server.diag     4.5.0.6  COMMITTED  ES Server Diags
  cluster.es.server.events   4.5.0.6  COMMITTED  ES Server Events
  cluster.es.server.rte      4.5.0.6  COMMITTED  ES Base Server Runtime
  cluster.es.server.utils    4.5.0.6  COMMITTED  ES Server Utilities
  cluster.license            4.5.0.2  COMMITTED  HACMP Electronic License
```

Enterprise Storage Server (ESS) filesets

```
{node4:root}/home/michel-> lslpp -lOu ibm*
  Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  ibm2105.rte             32.6.100.13  COMMITTED  IBM 2105 Disk Device
  ibmSdd_510nchacmp.rte     1.3.3.11  COMMITTED  IBM Subsystem Device Driver
                                                 AIX V51 for non-concurrent
                                                 HACMP
  ibmpfe.essutil.fibre.data  1.0.6.0  COMMITTED  IBM 2105 ESS Utilities Fibre
                                                 Attachment Support
  ibmpfe.essutil.rte         1.0.8.0  COMMITTED  IBM 2105 ESS Utilities Runtime
                                                 Environment
  ibmpfe.essutil.scsi.data   1.0.5.0  COMMITTED  IBM 2105 ESS Utilities SCSI
                                                 Attachment Support
```

Fibre Channel drivers filesets

```
{node4:root}/home/michel-> lslpp -lOu devices* |grep -i fc
  devices.common.IBM.fc.hba-api
                            5.2.0.10  COMMITTED  Common HBA API Library
  devices.common.IBM.fc.rte
                            5.2.0.10  COMMITTED  Common IBM FC Software
  devices.fcp.disk.array.diag
                            5.2.0.10  COMMITTED  Fibre Channel RAID Device
                                                 Diagnostics
  devices.fcp.disk.array.rte
                            5.2.0.10  COMMITTED  FC SCSI RAIDiant Array Device
```

```
  devices.fcp.disk.ibm2105.rte
                        32.6.100.13  COMMITTED  FCP 2105 Disk Device
  devices.fcp.disk.rte      5.2.0.10  COMMITTED  FC SCSI CD-ROM, Disk,
  devices.fcp.tape.rte      5.2.0.10  COMMITTED  FC SCSI Tape Device Software
  devices.pci.df1000f7.com  5.2.0.10  COMMITTED  Common PCI FC Adapter Device
  devices.pci.df1000f7.diag 5.2.0.10  COMMITTED  PCI FC Adapter Device
  devices.pci.df1000f7.rte  5.2.0.10  COMMITTED  PCI FC Adapter Device Software
  devices.pci.df1000f9.diag  5.2.0.0  COMMITTED  64-bit PCI FC Adapter Device
  devices.pci.df1000f9.rte  5.2.0.10  COMMITTED  64-bit PCI FC Adapter Device
```

Java filesets

```
{node1:oracle}/oracle/home-> lslpp -L |grep Java
  Java131.rte.bin            1.3.1.2    C     F    Java Runtime Environment
  Java131.rte.lib            1.3.1.2    C     F    Java Runtime Environment
  idebug.rte.hpj             9.2.5.0    C     F    High-Performance Java Runtime
  idebug.rte.jre             9.2.5.0    C     F    Java Runtime Environment
  idebug.rte.olt.Java        9.2.5.0    C     F    Object Level Trace Java
```

Troubleshooting

Check logs

If you encounter any application or system problem, you should check the logs of the failure area to identify the root cause. The following is a list of the logs of different components.

Hardware and AIX error report

- errpt

Group Service and Topology Service logs

- RSCT HACMP related:

  Topology Service: /var/ha/log/topsvcs.*

  Group Service: /var/ha/log/grpsvcs.*

- RSCT Peer Domain (RPD) related:

  Topolgy Service: /var/ct/<cluster_name>/log/cthats

  Group Service: /var/ct/<cluster_nam>/log/cthags

HACMP/ES logs

- /tmp/hacmp.out
- /tmp/clstrmgr.debug

GPFS logs

- /var/adm/ras/mmfs.log.latest

Oracle logs and trace

Identify the log location from the initialization parameter of

- background_dump_dest:

  The setting of this init.ora parameter contains the path name of a directory where debugging trace files of the background processes are written during the Oracle operations. An alert file “alert_<ORACLE_SID>.log” logs significant database events and messages.

- user_dump_dest

  This init.ora parameter contains the path name of a directory where Oracle writes debugging trace file for a user process.

AIX related problems

Is any file system full?

Execute ‘df’ and search 100% used.

Check Oracle datafile ownership and permission

Execute ’ls -al <datafiles>’ to make sure the datafile ownership and permission. Some of the Oracle creation failures are caused by the wrong file settings.

Network Related

Is the network up?

Since RAC, GPFS and HACMP/ES are all cluster oriented, if any network becomes unavailable, some components will fail. The following is the network dependency of different products.

- RAC has dependency of private cluster interconnects for inter-instance communications. It also needs a client to server network for client transactions to come in.
- GPFS has dependency on its private GPFS network to pass tokens.
- HACMP/ES which needs to monitor or perform failover for all the networks defined in the topology.

Use any of the following commands to verify the network status.

- ping <IP_address>

  If the network is up, you should see echo from the pinged IP address.

- ifconfig -a

  You should see `UP` for each network interface that is up.

- netstat -ni

  If the interface name has an asterisk (*) in front of the name listed in the **`netstat -ni`** command output, for example `*en4`, it means that the network interface is down.

HACMP check issues

What is the HACMP cluster status?

Execute **`issrc -g cluster`** to find out the status.

HACMP'cllsif' utility

Execute **`cllsif`** to show adapter information.

Heartbeat rate tuning

If you adjust the heartbeat rate too aggressively, although the node failover may be faster, but it may cause the false node down situation if you have a slow network. You should test your network before making any adjustment of the HACMP heartbeat rate.

GPFS check issues

How to verify the GPFS status?

```
{node1:root}/-> lssrc -s mmfs
Subsystem         Group            PID          Status
 mmfs             aixmm            454904       active
{node1:root}/-> mount|grep mmfs
         /dev/data        /data            mmfs   Jul 09 12:34 rw
         /dev/oracle      /oracle          mmfs   Jul 09 12:34 rw
```

Mount GPFS may be long for the first time - be patient

If this is the first time that a GPFS is mounted, be patient, GPFS may need to vary on the volume groups and disks belonging to the file system. Depends on the number of volume groups, it may take some time.

Oracle check issues

RAC is one database with multiple instances

One misconception is that RAC has multiple databases. Although RAC has many instances, it is only one database with multiple instances. An instance is a memory structure with a set of Oracle background processes. Normally each physical node has only one Oracle instance.

Upgrade to Oracle 9.2.0.3?

At the time when this redbook is published, the highest Oracle9i release is Oracle 9.2.0.3. If this is a new RAC creation, we recommend you apply the Oracle 9.2.0.3 patch immediately after the Oracle 9.2.0.1 installation, before creation the database.

If you encounter any Oracle problem and your current Oracle release is either at Oracle 9.2.0.1 or Oracle 9.2.0.2, we recommend you to search the Oracle 9.2.0.3 patch release notes for your bug description to find out whether there is a match

Why RAC installation does not have any RAC binaries?

You need to bring up HACMP cluster before the Oracle installation. If the HACMP is not up, you will not receive any warning from Oracle Universal Installer (OUI), and only Oracle SMP binaries will get installed. In additional to checking the HACMP cluster status, when you executed 'rootpre.sh', you should see the following message:

```
Adding r/w perms for group hagsuser to /var/ha/soc/grpsvcsdsocket.<cluster_ame>..
success.
```

If you do not see this message, you should verify your HACMP cluster status again. If your RAC installation is successful, you should find 'srvcctl', 'srvconfig', 'gsd', 'gsdctl' under the $ORACLE_HOME/bin.

How to cleanup shared memory after Oracle instance failure

If the Oracle instance startup fails, you should check the alert file and trace files under the path name specified by the 'background_dump_dest'. It should give you some indication why the Oracle instance startup fails.

If after executing 'shutdown abort', the Oracle shared memory segment still lingers, you can execute the

```
ipcrm -m <SharedMemoryID>
```

to clean up the Oracle related shared memory.

'srvctl' failure

If you execute the 'srvctl' command, e.g.,

```
srvctl start database -d <database_name>
```

and get the following error message:

```
PRKR-1007 : getting of cluster database <db_name> configuration failed,
PRKC-1019 : Error creating handle to daemon on the node <hostname>
PRKO-2005 : Application error: Failure in getting Cluster Database Configuration for:
<db_name>
```

This error may be caused by 'gsd' daemon is not up. To verify, you can execute 'ps -ef | grep gsd', or 'gsdctl stat'.

GSD and database creation

Global Services Daemon (GSD) is a process that receives requests from the ‘SRVCTL’ utility to execute administrative tasks such as startup or shutdown. The command is executed on each node and the results are returned to the SRVCTL. The GSD needs to be started on all nodes.

When created database failed. one of the most frequently asked questions is whether one needs to startup GSD first before the database creation. It is true that Database Configuration Assistant (DBCA) and NETCA have dependency on GSD, but this service daemon has nothing to do with the create database failure.If

Stop GSD first before executing ‘svrconfig -init’

To administer the RAC with SRVCTL, you need the Global Services Daemon (GSD) to be up. However, to configure the cluster-wide SRVM configuration, after you create or edit the file ‘/var/opt/oracle/srvConfig.loc’, and add the entry ‘srvconfig_loc=path_name’, you need to ensure that GSD is not running before you initialize the shared file.

```
srvconfig -init
```

Otherwise, the initialization will fail. To stop the GSD daemon, execute:

```
gsdctl stop
```

LD_LIBRARY_PATH for the graphical NETCA

You need to include the ‘$ORACLE_HOME/lib32’ as part of the LD_LIBRARY_PATH’ in the “~/.profile” of the Oracle user ID. Otherwise, the ‘NETCA’ will fail.

The shared ORACLE_HOME reminder

If used default listener name ‘LISTENER’, then the ‘listener.log’ may be shared among all instances.

AIX Tuning considerations

Using Asynchronous I/O

Oracle task full advantage of asynchronous I/O (AIO) provided by AIX, resulting in faster AIX access. AIO interleaves multiple I/O to improve I/O subsystem throughput. The advantage of AIO is realized when data is well distributed among different disks.

AIX version 4 and higher support asynchronous I/O (AIO) for database files created on both on file system and on raw devices. When using AIO on file systems, the kernel server processes (kproc) control each request from the time a request is taken off the queue until it completes.

Use the following commands to set the number of servers.

- smit aio
- chdev -l aio0 -a maxservers='m' -a 'minservers='n'

The default server for the minimum number is 1. The default value for the maximum value to the number of servers is 10. These value are usually too low to run the Oracle server on large system with 4 CPUs or more. Oracle recommends the following values:

- MINSERVERS: varies depending on the asynchronous requests to the AIO servers on the system. Oracle recommends an initial value equals to the number of CPUs or 10, whichever is lower. Once an appropriate MAXSERVERS value is determined, MINSERVERS can be set to (MAXSREVERS/2).

File system read ahead is not required for GPFS Direct I/O

Since GPFS supports direct I/O, there is no need to enable the file system read ahead. The read ahead should be enabled from Oracle by adjusting the Oracle initialization and tuning parameter, db_file_multiblock_read_count.

Disk queue_depth tuning

Tuning the disk command queue depth may be beneficial in certain environments. You do not have to perform this operation if you are using IBM disks, because the AIX device drivers recognize the devices and configure the proper parameters.

If you have non-IBM disks (SCSI), you have to check with the disk manufacturer if the queue depth parameter is supported, and find out the size of the queue. then, you can set it using the following command:

```
chdev -l hdisk<x> -a queue_depth=<value>
```

AIX Performance Monitoring Tool

Beside the system tools for performance problem determination (vmstat, sar, svmon, iostat, netstat, tcpdump, entstat etc.,), you can also use the following monitoring tools for AIX performance:

- Topas

  In AIX5, this tool comes in bos.perf.tools package. This tool can be used to provide a full screen of a variety of performance statistics.

- nmon

  Is a free tool to analyze AIX performance and can be downloaded from:

  `http://www-1.ibm.com/servers/esdd/articles/analyze_aix/index.html`

B

Troubleshooting

This appendix provides suggested actions you can take to troubleshoot application or system problems you may encounter.

Check logs

If you encounter any application or system problem, you should check the logs of the failure area to identify the root cause. The following is a list of the logs of different components.

Hardware and AIX error report

- errpt

Group Service and Topology Service logs

- RSCT HACMP-related:

  Topology Service: /var/ha/log/topsvcs.*

  Group Service: /var/ha/log/grpsvcs.*

- RSCT Peer Domain (RPD)-related:

  Topology Service: /var/ct/<cluster_name>/log/cthats

  Group Service: /var/ct/<cluster_nam>/log/cthags

HACMP/ES logs

- /tmp/hacmp.out
- /tmp/clstrmgr.debug

GPFS logs

- /var/adm/ras/mmfs.log.latest

Oracle logs and trace

Identify the log location from the initialization parameter of:

- background_dump_dest

  The setting of this init.ora parameter contains the path name of a directory where debugging trace files of the background processes are written during the Oracle operations. An alert file named alert_<ORACLE_SID>.log logs significant database events and messages.

- user_dump_dest

  This init.ora parameter contains the path name of a directory where Oracle writes debugging trace file for a user process.

AIX-related problems

Is any file system full?

Execute **`df`** and search 100% used.

Check Oracle datafile ownership and permission

Execute **`ls -al <datafiles>`** to verify the datafile ownership and permission. Some of the Oracle creation failures are caused by the wrong file settings.

Network-related problems

Is the network up?

Since RAC, GPFS and HACMP/ES are all cluster-oriented, and if any network becomes unavailable, some components will fail. The following is the network dependency of different products.

- RAC has a dependency of private cluster interconnects for inter-instance communications. It also needs a client-to-server network for client transactions to come in.
- GPFS has a dependency on its private GPFS network to pass tokens.
- HACMP/ES needs to monitor or perform failover for all the networks defined in the topology.

Use any of the following commands to verify the network status.

- `ping <IP_address>`

  If the network is up, you should see an echo from the pinged IP address.

- `ifconfig -a`

  You should see `UP` for each network interface that is up.

- `netstat -ni`

  If the interface name has an asterisk (*) at the end of the name, for example `er4*`, it means that the network is down.

HACMP check issues

What is the HACMP cluster status?

Execute **`issrc -g cluster`** to find out the status.

HACMP cllsif utility

Execute **`cllsif`** to show adapter information.

Heartbeat rate tuning

If you adjust the heartbeat rate too aggressively, the node failover may be faster, but it may cause the false node down situation if you have a slow network. You should test your network before making any adjustment of the HACMP heartbeat rate.

GPFS check issues

How to verify GPFS status

```
{node1:root}/-> lssrc -s mmfs
Subsystem         Group            PID          Status
 mmfs             aixmm            454904       active
{node1:root}/-> mount|grep mmfs
         /dev/data        /data            mmfs   Jul 09 12:34 rw
         /dev/oracle      /oracle          mmfs   Jul 09 12:34 rw
```

Mount GPFS may take a long initially - be patient

If this is the first time that a GPFS is mounted, be patient, GPFS may need to vary on the volume groups and disks belonging to the file system. Depending on the number of volume groups, this may take some time.

Oracle check issues

RAC is one database with multiple instances

It is a misconception that RAC has multiple databases. Although RAC has many instances, it is only one database with multiple instances. An *instance* is a memory structure with a set of Oracle background processes. Normally each physical node has only one Oracle instance.

Should I upgrade to Oracle 9.2.0.3?

At the time of writing, the highest Oracle9*i* release is Oracle 9.2.0.3. If this is a new RAC creation, we recommend that you apply the Oracle 9.2.0.3 patch immediately after the Oracle 9.2.0.1 installation, before creating the database.

If you encounter any Oracle problem and your current Oracle release is either at Oracle 9.2.0.1 or Oracle 9.2.0.2, we recommend that you search the Oracle 9.2.0.3 patch release notes to see if there is a description of your bug.

Why doesn't my RAC installation have any RAC binaries?

You need to bring up the HACMP cluster before the Oracle installation. If the HACMP is not up, you will not receive any warning from Oracle Universal Installer (OUI), and only Oracle SMP binaries will get installed. In additional to checking the HACMP cluster status, when you execute **`rootpre.sh`**, you should see the following message:

```
Adding r/w perms for group hagsuser to /var/ha/soc/grpsvcsdsocket.<cluster_ame>..
success.
```

If you do not see this message, you should verify your HACMP cluster status again. If your RAC installation is successful, you should find srvctl, srvconfig, gsd, and gsdctl in the $ORACLE_HOME/bin directory.

How to clean up shared memory after an Oracle instance failure

If the Oracle instance startup fails, you should check the alert file and trace files under the path name specified by background_dump_dest; this should give you some indication why the Oracle instance startup fails.

After executing shutdown abort, if the Oracle shared memory segment still lingers, you can execute the following to clean up the Oracle-related shared memory:

```
ipcrm -m <SharedMemoryID>
```

srvctl failure

If you execute the **`srvctl`** command, for example:

```
srvctl start database -d <database_name>
```

you may get the following error message:

```
PRKR-1007 : getting of cluster database <db_name> configuration failed,
PRKC-1019 : Error creating handle to daemon on the node <hostname>
PRKO-2005 : Application error: Failure in getting Cluster Database Configuration for:
<db_name>
```

This error may occur because the gsd daemon is not up. To verify, you can execute **`ps -ef | grep gsd`**, or **`gsdctl stat`**.

GSD and database creation

Global Services Daemon (GSD) is a process that receives requests from the **srvctl** utility to execute administrative tasks such as startup or shutdown. The command is executed on each node and the results are returned to the SRVCTL. The GSD needs to be started on all nodes.

When database creation failed, one of the most frequently asked questions is whether you need to start up GSD first, before database creation. Although it is true that Database Configuration Assistant (DBCA) and NETCA have dependencies on GSD, this service daemon has nothing to do with the database creation failure.

Stop GSD first before executing svrconfig -init

To administer the RAC with **srvctl**, the Global Services Daemon (GSD) needs to be up. However, to configure the cluster-wide SRVM configuration, after you create or edit the file /var/opt/oracle/srvConfig.loc, and add the entry `srvconfig_loc=path_name`, you need to ensure that GSD is *not* running before you initialize the shared file, as follows:

```
srvconfig -init
```

Otherwise, the initialization will fail. To stop the GSD daemon, execute:

```
gsdctl stop
```

LD_LIBRARY_PATH for the graphical NETCA

You need to include $ORACLE_HOME/lib32 as part of the LD_LIBRARY_PATH in the ~/.profile of the oracle user ID. Otherwise, NETCA will fail.

The shared ORACLE_HOME reminder

If you used the default listener name LISTENER, then the listener.log may be shared among all instances.

AIX tuning considerations

Using asynchronous I/O

Oracle takes full advantage of asynchronous I/O (AIO) provided by AIX, resulting in faster AIX access. AIO interleaves multiple I/O to improve I/O subsystem throughput. The advantage of AIO is realized when data is well distributed among different disks.

AIX version 4 and higher support asynchronous I/O (AIO) for database files created both on file system and on raw devices. When using AIO on file systems, the kernel server processes (kproc) control each request from the time a request is taken off the queue until it completes.

Use the following commands to set the number of servers:

- `smit aio`
- `chdev -l aio0 -a maxservers='m' -a 'minservers='n'`

The default server for the minimum number is 1. The default value for the maximum value to the number of servers is 10. These value are usually too low to run the Oracle server on large systems with four or more CPUs. Oracle recommends the following values:

MINSERVERS - this value varies, depending on the asynchronous requests to the AIO servers on the system. Oracle recommends an initial value equal to the number of CPUs or 10, whichever is lower. Once an appropriate MAXSERVERS value is determined, MINSERVERS can be set to (MAXSREVERS/2).

File system read ahead is not required for GPFS direct I/O

Since GPFS supports direct I/O, there is no need to enable the file system read ahead. The read ahead should be enabled from Oracle by adjusting the Oracle initialization and tuning parameter db_file_multiblock_read_count.

Disk queue_depth tuning

Tuning the disk command queue depth may be beneficial in certain environments. However, you do not have to perform this operation if you are using IBM disks, because the AIX device drivers recognize the devices and configure the proper parameters.

If you have non-IBM disks (SCSI), you have to check with the disk manufacturer if the queue depth parameter is supported, and find out the size of the queue. Then you can set this parameter using the following command:

```
chdev -l hdisk<x> -a queue_depth=<value>
```

AIX Performance Monitoring Tool

In addition to using the system tools for performance problem determination (vmstat, sar, svmon, iostat, netstat, tcpdump, entstat, and so on), you can also use the following monitoring tools for AIX performance:

- Topas

  In AIX5, this tool is in the bos.perf.tools package. It can be used to provide a full screen of a variety of performance statistics.

- nmon

  This is a free tool to analyze AIX performance; it can be downloaded from:

  ```
  http://www-1.ibm.com/servers/esdd/articles/analyze_aix/index.html
  ```

C

HACMP cluster configuration output

This appendix provides the output of a valid HACMP configuration that you can use to check against your own configuration.

Two interesting commands are discussed:

- **`cllscf`** is the cluster description.
- **`cllsif`** shows the networks and interfaces.

Following is a description of the network names and functions:

- ora_interconnect1 and ora_interconnect2 are the networks used for the Oracle9*i* RAC cache fusion.
- sernet1_2 is the serial network between node1 and node2, for HACMP heartbeats.
- The remaining sernet networks follow the same pattern.

Figure C-1 on page 222 shows the Oracle interconnect private networks defined in the HACMP cluster configuration.

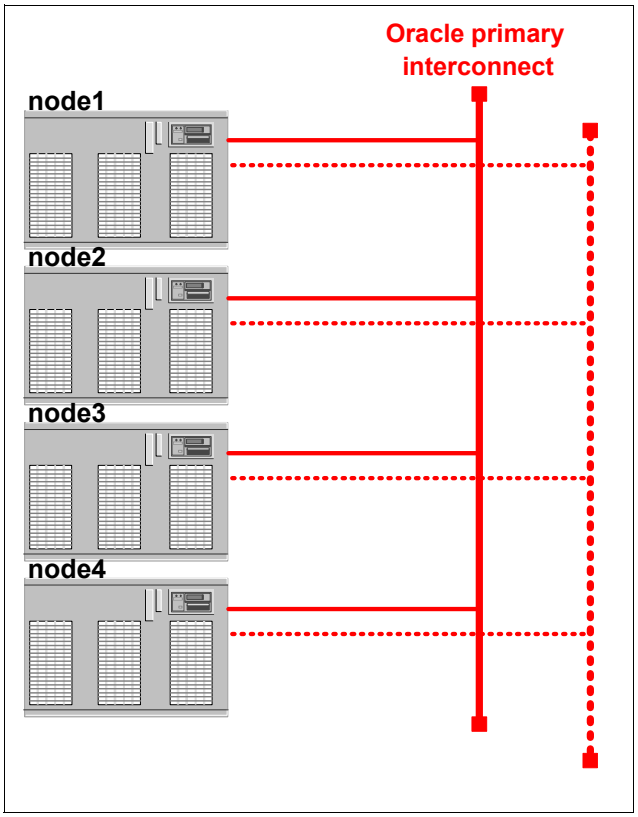


Figure C-1 HACMP private networks for Oracle9i RAC interconnect

Figure C-2 displays the topology of the serial networks for a 4-node cluster. In a 2-node architecture, only one serial network is required.

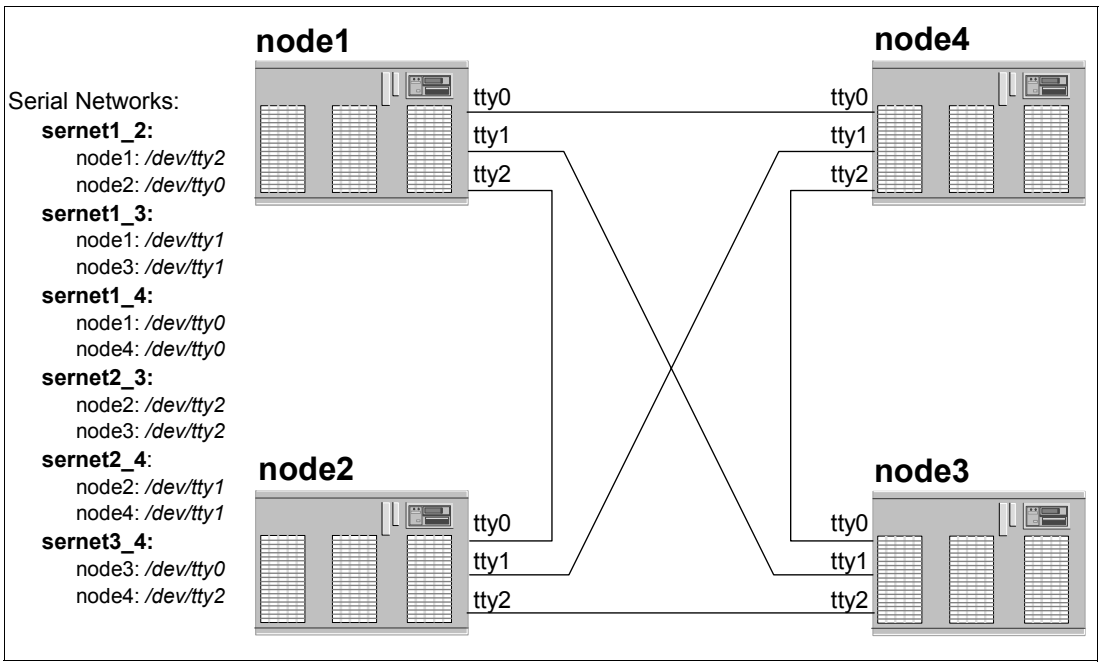


Figure C-2 HACMP serial networks topology for heart beats

Cluster description (cllscf command)

This output is displayed when you use the following **smit hacmp** screen:

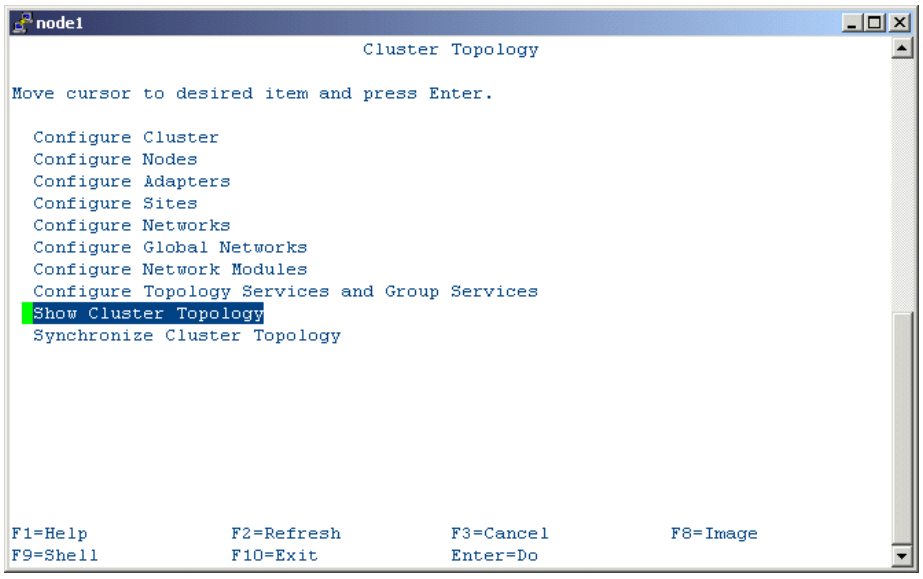


Figure C-3 Output of smit hacmp

The same output can also be displayed with the **cllscf** command, as shown in Example C-1.

The warning (INVALID) is not a problem. In the field, HACMP is mainly used for IP address takeover. In this case, a service address must be associated with a boot address, and also a standby one. But because we are not using any kind of IPAT with Oracle9*i* RAC, this warning is not important.

Example: C-1 HACMP cllscf command output

```
{node1:root}/-> /usr/es/sbin/cluster/utilities/cllscf

Cluster Description of Cluster oracle9irac
Cluster ID: 1000
Cluster Security Level Standard
There were 8 networks defined: ora_interconnect1, ora_interconnect2, sernet1_2, sernet1_3,
sernet1_4, sernet2_3, sernet2_4, sernet3_4
There are 4 nodes in this cluster

NODE node1:
        This node has 5 service interface(s):

        Service Interface onet11:
                IP address:     10.10.10.61
                Hardware Address:
                Network:        ora_interconnect1
                Attribute:      private
                Aliased Address?:       Not Supported

        (INVALID) Service Interface onet11 has no boot interfaces
        Service Interface onet11 has no standby interfaces


        Service Interface onet21:
```

```
        IP address:     172.16.10.61
        Hardware Address:
        Network:        ora_interconnect2
        Attribute:      private
        Aliased Address?:       Not Supported

    (INVALID) Service Interface onet21 has no boot interfaces
    Service Interface onet21 has no standby interfaces


    Service Interface node1_serial2:
        IP address:     /dev/tty2
        Hardware Address:
        Network:        sernet1_2
        Attribute:      serial
        Aliased Address?:       Not Supported

    (INVALID) Service Interface node1_serial2 has no boot interfaces
    Service Interface node1_serial2 has no standby interfaces


    Service Interface node1_serial1:
        IP address:     /dev/tty1
        Hardware Address:
        Network:        sernet1_3
        Attribute:      serial
        Aliased Address?:       Not Supported

    (INVALID) Service Interface node1_serial1 has no boot interfaces
    Service Interface node1_serial1 has no standby interfaces


    Service Interface node1_serial0:
        IP address:     /dev/tty0
        Hardware Address:
        Network:        sernet1_4
        Attribute:      serial
        Aliased Address?:       Not Supported

    (INVALID) Service Interface node1_serial0 has no boot interfaces
    Service Interface node1_serial0 has no standby interfaces


NODE node2:
    This node has 5 service interface(s):

    Service Interface onet12:
        IP address:     10.10.10.62
        Hardware Address:
        Network:        ora_interconnect1
        Attribute:      private
        Aliased Address?:       Not Supported

    (INVALID) Service Interface onet12 has no boot interfaces
    Service Interface onet12 has no standby interfaces


    Service Interface onet22:
        IP address:     172.16.10.62
        Hardware Address:
```

```
                Network:        ora_interconnect2
                Attribute:      private
                Aliased Address?:       Not Supported

        (INVALID) Service Interface onet22 has no boot interfaces
        Service Interface onet22 has no standby interfaces


        Service Interface node2_serial0:
                IP address:     /dev/tty0
                Hardware Address:
                Network:        sernet1_2
                Attribute:      serial
                Aliased Address?:       Not Supported

        (INVALID) Service Interface node2_serial0 has no boot interfaces
        Service Interface node2_serial0 has no standby interfaces


        Service Interface node2_serial2:
                IP address:     /dev/tty2
                Hardware Address:
                Network:        sernet2_3
                Attribute:      serial
                Aliased Address?:       Not Supported

        (INVALID) Service Interface node2_serial2 has no boot interfaces
        Service Interface node2_serial2 has no standby interfaces


        Service Interface node2_serial1:
                IP address:     /dev/tty1
                Hardware Address:
                Network:        sernet2_4
                Attribute:      serial
                Aliased Address?:       Not Supported

        (INVALID) Service Interface node2_serial1 has no boot interfaces
        Service Interface node2_serial1 has no standby interfaces


NODE node3:
        This node has 5 service interface(s):

        Service Interface onet13:
                IP address:     10.10.10.63
                Hardware Address:
                Network:        ora_interconnect1
                Attribute:      private
                Aliased Address?:       Not Supported

        (INVALID) Service Interface onet13 has no boot interfaces
        Service Interface onet13 has no standby interfaces


        Service Interface onet23:
                IP address:     172.16.10.63
                Hardware Address:
                Network:        ora_interconnect2
                Attribute:      private
```

```
                Aliased Address?:       Not Supported

        (INVALID) Service Interface onet23 has no boot interfaces
        Service Interface onet23 has no standby interfaces


        Service Interface node3_serial1:
                IP address:     /dev/tty3
                Hardware Address:
                Network:        sernet1_3
                Attribute:      serial
                Aliased Address?:       Not Supported

        (INVALID) Service Interface node3_serial1 has no boot interfaces
        Service Interface node3_serial1 has no standby interfaces


        Service Interface node3_serial2:
                IP address:     /dev/tty2
                Hardware Address:
                Network:        sernet2_3
                Attribute:      serial
                Aliased Address?:       Not Supported

        (INVALID) Service Interface node3_serial2 has no boot interfaces
        Service Interface node3_serial2 has no standby interfaces


        Service Interface node3_serial0:
                IP address:     /dev/tty0
                Hardware Address:
                Network:        sernet3_4
                Attribute:      serial
                Aliased Address?:       Not Supported

        (INVALID) Service Interface node3_serial0 has no boot interfaces
        Service Interface node3_serial0 has no standby interfaces


NODE node4:
        This node has 5 service interface(s):

        Service Interface onet14:
                IP address:     10.10.10.64
                Hardware Address:
                Network:        ora_interconnect1
                Attribute:      private
                Aliased Address?:       Not Supported

        (INVALID) Service Interface onet14 has no boot interfaces
        Service Interface onet14 has no standby interfaces


        Service Interface onet24:
                IP address:     172.16.10.64
                Hardware Address:
                Network:        ora_interconnect2
                Attribute:      private
                Aliased Address?:       Not Supported
```

```
        (INVALID) Service Interface onet24 has no boot interfaces
        Service Interface onet24 has no standby interfaces


        Service Interface node4_serial0:
                IP address:     /dev/tty0
                Hardware Address:
                Network:        sernet1_4
                Attribute:      serial
                Aliased Address?:       Not Supported

        (INVALID) Service Interface node4_serial0 has no boot interfaces
        Service Interface node4_serial0 has no standby interfaces


        Service Interface node4_serial1:
                IP address:     /dev/tty1
                Hardware Address:
                Network:        sernet2_4
                Attribute:      serial
                Aliased Address?:       Not Supported

        (INVALID) Service Interface node4_serial1 has no boot interfaces
        Service Interface node4_serial1 has no standby interfaces


        Service Interface node4_serial2:
                IP address:     /dev/tty2
                Hardware Address:
                Network:        sernet3_4
                Attribute:      serial
                Aliased Address?:       Not Supported

        (INVALID) Service Interface node4_serial2 has no boot interfaces
        Service Interface node4_serial2 has no standby interfaces




Breakdown of network connections:

Connections to network ora_interconnect1
        Node node1 is connected to network ora_interconnect1 by these interfaces:
                onet11

        Node node2 is connected to network ora_interconnect1 by these interfaces:
                onet12

        Node node3 is connected to network ora_interconnect1 by these interfaces:
                onet13

        Node node4 is connected to network ora_interconnect1 by these interfaces:
                onet14


Connections to network ora_interconnect2
        Node node1 is connected to network ora_interconnect2 by these interfaces:
                onet21

        Node node2 is connected to network ora_interconnect2 by these interfaces:
```

```
                onet22

        Node node3 is connected to network ora_interconnect2 by these interfaces:
                onet23

        Node node4 is connected to network ora_interconnect2 by these interfaces:
                onet24


Connections to network sernet1_2
        Node node1 is connected to network sernet1_2 by these interfaces:
                node1_serial2

        Node node2 is connected to network sernet1_2 by these interfaces:
                node2_serial0


Connections to network sernet1_3
        Node node1 is connected to network sernet1_3 by these interfaces:
                node1_serial1

        Node node3 is connected to network sernet1_3 by these interfaces:
                node3_serial1


Connections to network sernet1_4
        Node node1 is connected to network sernet1_4 by these interfaces:
                node1_serial0

        Node node4 is connected to network sernet1_4 by these interfaces:
                node4_serial0


Connections to network sernet2_3
        Node node2 is connected to network sernet2_3 by these interfaces:
                node2_serial2

        Node node3 is connected to network sernet2_3 by these interfaces:
                node3_serial2


Connections to network sernet2_4
        Node node2 is connected to network sernet2_4 by these interfaces:
                node2_serial1

        Node node4 is connected to network sernet2_4 by these interfaces:
                node4_serial1


Connections to network sernet3_4
        Node node3 is connected to network sernet3_4 by these interfaces:
                node3_serial0

        Node node4 is connected to network sernet3_4 by these interfaces:
                node4_serial2
```

Cluster networks (cllsif command)

The networks and adapters configured in the HACMP cluster can be displayed by the **cllsif** command.

Note that this output shows all the networks defined for our four nodes Oracle9*i* RAC cluster definition. In your configuration, you can have less serial networks, depending on your actual number of nodes.

The output shown in Example C-2 has been reformatted for better readability.

Example: C-2 HACMP cllsif command output

```
{node1:root}/-> /usr/es/sbin/cluster/utilities/cllsif

Adapter       Type       Network           Net Type   Attribute  Node   IPaddress Interface

onet11        service    ora_interconnect1 ether      private    node1  10.10.10.61    en1
onet21        service    ora_interconnect2 ether      private    node1  172.16.10.61   en2
node1_serial0 service    sernet1_4         rs232      serial     node1  /dev/tty0
node1_serial1 service    sernet1_3         rs232      serial     node1  /dev/tty1
node1_serial2 service    sernet1_2         rs232      serial     node1  /dev/tty2

onet12        service    ora_interconnect1 ether      private    node2  10.10.10.62    en1
onet22        service    ora_interconnect2 ether      private    node2  172.16.10.62   en2
node2_serial0 service    sernet1_2         rs232      serial     node2  /dev/tty0
node2_serial1 service    sernet2_4         rs232      serial     node2  /dev/tty1
node2_serial2 service    sernet2_3         rs232      serial     node2  /dev/tty2

onet13        service    ora_interconnect1 ether      private    node3  10.10.10.63    en1
onet23        service    ora_interconnect2 ether      private    node3  172.16.10.63   en2
node3_serial0 service    sernet3_4         rs232      serial     node3  /dev/tty0
node3_serial1 service    sernet1_3         rs232      serial     node3  /dev/tty3
node3_serial2 service    sernet2_3         rs232      serial     node3  /dev/tty2

onet14        service    ora_interconnect1 ether      private    node4  10.10.10.64    en1
onet24        service    ora_interconnect2 ether      private    node4  172.16.10.64   en2
node4_serial0 service    sernet1_4         rs232      serial     node4  /dev/tty0
node4_serial1 service    sernet2_4         rs232      serial     node4  /dev/tty1
node4_serial2 service    sernet3_4         rs232      serial     node4  /dev/tty2
```

Some useful AIX commands

AIX

- **lslpp -l** - List software products
- **aioo -a** - List the asynchronous servers.
- **vmo -L** - List the VMM tuning parameters
- **svmon -G** - Snapshot of the current virtual memory
- I/O related: **lspv**, **lsvg**
- File system-related: **mount**, **df**

Network

- **netstat -ni** - Show network status

EtherChannel

- **`lsattr -El <EtherChannel adapter>`**, for example, **`lsattr -El ent4`**

  Shows all the network adapter names under the EtherChannel. The mode field should report `round_robin` if you configure EtherChannel this way.

- **`lsdev -Cc adapter`**

  Lists all the adapters. EtherChannel is reported. This also includes all the interfaces used in the EtherChannel.

- entstat -d

  Shows Ethernet device driver and device statistics.

HACMP

- **`lssrc -g cluster`** - Can be used to verify whether the HACMP cluster subsystems are active.
- /usr/es/sbin/cluster/utilities/cllsif - Shows all the adapter information.
- **`lssrc -ls topsvcs`** - Lists whether the topology service is active.
- **`lssrc -ls grpsvcs`** - Lists whether group service is active.

GPFS

- **`mmlscluster`** - Displays the current configuration for a GPFS cluster. It shows the cluster ID, primary repository server, backup repository server, and nodes in the nodeset.
- **`mmlsconfig`** - Displays the current configuration data for a given nodeset. It shows the GPFS cluster type and the pagepool size.
- **`mmchfs`** - Changes the attributes of a GPFS file system.

Oracle

srvctl

srvctl startlstop database -d <db_name>

EtherChannel setup procedures

1. Configure the EtherChannel interface.
   - **`smitty`**, then select **Devices** -> **Communications** -> **EtherChannel**.
     - Add an EtherChannel and select the network adapters (for example, ent0, ent2, ent3, or ent5).
     - Enable Round Robin mode: **Yes**.
     - Enable Gigabit Ethernet Jumbo frame: **Yes**.
2. Assign an IP address to the EtherChannel interface created in the previous step.
   - **`smitty tcpip`**, then select **Further configuration** -> **Network Interface** -> **Network Interface Selection**.

     Change or show characteristics of a network interface (for example, pick ent6, which is the EtherChannel interface defined).

     Note: Make sure that the Internet address and netmasks are correct.

EtherChannel test scenarios

EtherChannel satisfies the large bandwidth RAC interconnect requirement

An EtherChannel with multiple links is extremely beneficial for a RAC cluster with large SMP nodes, such as p690 with 32 ways. Although multiple gigabit networks can be connected between large SMP nodes without using EtherChannel as an interconnect, only one network link can be used by RAC as the private interconnect at any one time. The remaining network links can only be used as standbys for failover purposes. That is, as long as the primary RAC interconnect is still alive, the rest of the backup networks will never be used.

The EtherChannel round robin algorithm aggregates all the link bandwidths and satisfies the large bandwidth requirement problem.

RAC can use EtherChannel if specified under cluster_interconnects

If the IP address of the EtherChannel is specified as the setting of init.ora cluster_interconnects for each instance, RAC will use the EtherChannel as the designated cluster interconnect. RAC will use the specified EtherChannel regardless of whether or not the EtherChannel is configured and monitored under HACMP.

EtherChannel disconnects TNFF

Once the EtherChannel is specified as the setting of init.ora cluster_interconnects, it disables the Oracle Transparent Network Failover Failback (TNFF). That is, if the EtherChannel becomes unavailable, it will not fail over to the public client network.

An EtherChannel configured with multiple links has built-in high availability. As long as there one link is available, the Ethernet will continue to function.

In our test environment, we had a 2-link EtherChannel. When we disconnected one of the gigabit links under cache fusion traffic, EtherChannel stayed up and so did the RAC instances. The link failure is only reported in the AIX error report errpt.

EtherChannel with more than 2 links may provide better availability than TNFF

If RAC has to fail over its private interconnects to a public network, it cannot have good performance. That is, the Oracle client-to-server traffic is now intermixed with the RAC interconnect traffic.

An EtherChannel with multiple links using a round robin algorithm and aggregate network bandwidth provides better network performance and availability than the two private interconnects and one public network scheme. To verify the physical link activity, use:

```
entstat -d
```

D

Oracle9i RAC configuration files and sample scripts

In this appendix we show examples of Oracle9*i* RAC configuration files and sample scripts, and also discuss Oracle tuning considerations.

D.1 Sample of initialization parameters file

Example D-1 shows the initializing parameters init.ora used for our ITSO test database.

Example: D-1 init.ora

```
#*.aq_tm_processes=1
*.background_dump_dest='/oracle/admin/rac/bdump'
*.cluster_database_instances=4
*.cluster_database=true
*.compatible='9.2.0.0.0'
*.control_files='/data/oradata/rac/control01.ctl','/data/oradata/rac/control02.ctl','/data/
oradata/rac/control03.ctl'
*.core_dump_dest='/oracle/admin/rac/cdump'
*.db_block_size=16384
*.db_cache_size=536870912
*.db_domain=''
*.db_file_multiblock_read_count=16
*.db_name='rac'
*.sga_max_size=2602012616
#*.dispatchers='(PROTOCOL=TCP) (SERVICE=racXDB)'
*.fast_start_mttr_target=300
*.hash_join_enabled=TRUE
*.instance_name='rac'
rac3.instance_name='rac3'
rac2.instance_name='rac2'
rac1.instance_name='rac1'
rac4.instance_name='rac4'
rac4.instance_number=4
rac3.instance_number=3
rac1.instance_number=1
```

```
rac2.instance_number=2
*.java_pool_size=115343360
*.job_queue_processes=10
*.large_pool_size=67108864
*.open_cursors=300
*.pga_aggregate_target=134217728
*.processes=150
*.query_rewrite_enabled='FALSE'
*.remote_login_passwordfile='exclusive'
*.shared_pool_size=536870912
*.sort_area_size=524288
*.star_transformation_enabled='FALSE'
rac1.thread=1
rac2.thread=2
rac3.thread=3
rac4.thread=4
*.timed_statistics=TRUE
*.undo_management='AUTO'
*.undo_retention=10800
#*.undo_tablespace='UNDOTBS1'
rac2.undo_tablespace='UNDOTBS2'
rac1.undo_tablespace='UNDOTBS1'
rac3.undo_tablespace='UNDOTBS3'
rac4.undo_tablespace='UNDOTBS4'
*.user_dump_dest='/oracle/admin/rac/udump'
nls_date_format='dd mon yyyy hh24:mi:ss'
```

D.2 Sample database creation script

Create database script

Example D-2 shows the create database script CreateDB.sql. This script shows the database creation with a system tablespace, one thread of redo logs, and an undo tablespace for the RAC instance. /data is our GPFS database file system.

Example: D-2 CreateDB.sql

```
connect SYS/change_on_install as SYSDBA
set echo on
spool /oracle/product/9.2.0/assistants/dbca/logs/CreateDB.log

startup nomount pfile="/data/admin/rac/scripts/init.ora";

CREATE DATABASE rac
MAXINSTANCES 32
MAXLOGHISTORY 0
MAXLOGFILES 192
MAXLOGMEMBERS 3
MAXDATAFILES 1024
DATAFILE '/data/oradata/rac/system01.dbf' SIZE 512M REUSE AUTOEXTEND ON NEXT  1024OK
MAXSIZE UNLIMITED
EXTENT MANAGEMENT LOCAL
DEFAULT TEMPORARY TABLESPACE TEMP TEMPFILE '/data/oradata/rac/temp01.dbf' SIZE 4096M REUSE
AUTOEXTEND ON NEXT  640K MAXSIZE UNLIMITED
UNDO TABLESPACE "UNDOTBS1" DATAFILE '/data/oradata/rac/undotbs01.dbf' SIZE 4096M REUSE
AUTOEXTEND ON NEXT  512OK MAXSIZE UNLIMITED
CHARACTER SET WE8ISO8859P1
NATIONAL CHARACTER SET AL16UTF16
LOGFILE GROUP 1 ('/oracle/oradata/rac/rac_redo1_1.log') SIZE 1024OOK REUSE,
```

```
GROUP 2 ('/oracle/oradata/rac/rac_redo1_2.log') SIZE 102400K REUSE;

spool off
exit;
```

Create datafiles script

After the database is created, we use the following CreateDBfiles.sql script to create all the tablespaces and undo tablespaces for the rest of three RAC instances. /data is our GPFS database file system.

Example: D-3 CreateDbFiles.sql

```
connect SYS/change_on_install as SYSDBA
set echo on
spool /oracle/product/9.2.0/assistants/dbca/logs/CreateDBFiles.log

CREATE TABLESPACE "EXAMPLE" LOGGING DATAFILE '/data/oradata/rac/example01.dbf' SIZE 256M
REUSE AUTOEXTEND ON NEXT  640K MAXSIZE UNLIMITED EXTENT MANAGEMENT LOCAL SEGMENT SPACE
MANAGEMENT  AUTO ;

CREATE TABLESPACE "INDX" LOGGING DATAFILE '/data/oradata/rac/indx01.dbf' SIZE 4096M REUSE
AUTOEXTEND ON NEXT  1280K MAXSIZE UNLIMITED EXTENT MANAGEMENT LOCAL SEGMENT SPACE
MANAGEMENT  AUTO ;

CREATE TABLESPACE "TOOLS" LOGGING DATAFILE '/data/oradata/rac/tools01.dbf' SIZE 100M REUSE
AUTOEXTEND ON NEXT  320K MAXSIZE UNLIMITED EXTENT MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT
AUTO ;

CREATE UNDO TABLESPACE "UNDOTBS2" DATAFILE '/data/oradata/rac/undotbs02.dbf' SIZE 4096M
REUSE AUTOEXTEND ON NEXT  5120K MAXSIZE UNLIMITED;

CREATE UNDO TABLESPACE "UNDOTBS3" DATAFILE '/data/oradata/rac/undotbs03.dbf' SIZE 4096M
REUSE AUTOEXTEND ON NEXT  5120K MAXSIZE UNLIMITED;

CREATE UNDO TABLESPACE "UNDOTBS4" DATAFILE '/data/oradata/rac/undotbs04.dbf' SIZE 4096M
REUSE AUTOEXTEND ON NEXT  5120K MAXSIZE UNLIMITED;

CREATE TABLESPACE "USERS" LOGGING DATAFILE '/data/oradata/rac/users01.dbf' SIZE 512M REUSE
AUTOEXTEND ON NEXT  1280K MAXSIZE UNLIMITED EXTENT MANAGEMENT LOCAL SEGMENT SPACE
MANAGEMENT  AUTO ;

spool off
exit;
```

Post-database creation script

The PostDBCreation.sql script adds in the threads of redo logs for instance 2, 3 and 4. The GPFS is the database file system.

Example: D-4 PostDBCreation.sql

```
connect SYS/change_on_install as SYSDBA
set echo on
spool /oracle/product/9.2.0/assistants/dbca/logs/postDBCreation.log
@/oracle/product/9.2.0/rdbms/admin/utlrp.sql;
shutdown ;
```

```
connect SYS/change_on_install as SYSDBA
set echo on
spool /oracle/product/9.2.0/assistants/dbca/logs/postDBCreation.log

create spfile='/oracle/product/9.2.0/dbs/spfilerac.ora' FROM
pfile='/data/admin/rac/scripts/init.ora';

startup ;

ALTER DATABASE ADD LOGFILE THREAD 2 GROUP 3 ('/oracle/oradata/rac/rac_redo2_1.log') SIZE
102400K REUSE,
 GROUP 4 ('/oracle/oradata/rac/rac_redo2_2.log') SIZE 102400K REUSE;
ALTER DATABASE ENABLE PUBLIC THREAD 2;

ALTER DATABASE ADD LOGFILE THREAD 3 GROUP 5 ('/oracle/oradata/rac/rac_redo3_1.log') SIZE
102400K REUSE,
 GROUP 6 ('/oracle/oradata/rac/rac_redo3_2.log') SIZE 102400K REUSE;
ALTER DATABASE ENABLE PUBLIC THREAD 3;

ALTER DATABASE ADD LOGFILE THREAD 4 GROUP 7 ('/oracle/oradata/rac/rac_redo4_1.log') SIZE
102400K REUSE,
 GROUP 8 ('/oracle/oradata/rac/rac_redo4_2.log') SIZE 102400K REUSE;
ALTER DATABASE ENABLE PUBLIC THREAD 4;
spool off
shutdown ;
```

Oracle tuning considerations

Selecting the database block size

In the GPFS environment, we recommend 16 KB DB_BLOCK_SIZE for the Online Transaction Processing (OLTP) environment and 32 KB for the data warehouse environment.

Oracle monitoring tools

The following is a list of the monitoring tools provided by Oracle.

STATPACK

The STATPACK utility is the next generation of the utlbstat/utlestat report. It can be used to gather performance statistics and to detect performance problems. It enables the utlbstat/utlestat scripts to collect more data, including high resource SQL, and precalculates some ratios (such as cache hit ratio, per transaction and per second statistics). It also keeps a permanent repository, which makes historical data comparisons easier. It also separates data collection from report generation.

utlbstat/utlestat

Bstat/Estat is a set of SQL scripts under the $ORACLE_HOME/rdbms/admin directory:

- utlbstat.sql

  Creates a set of tables in your sys account, which contains a beginning snapshot of the database performance statistics.

- utlestat.sql

  Creates a second snapshot of these views and reports on the differences between the two snapshots to the file report.txt.

Initialization parameters

Using the DB_FILE_MULTIBLOCK_READ_COUNT parameter

A large value of DB_FILE_MULTIBLOCK_READ_COUNT usually yields better I/O throughput, especially for a sequential read environment. Set this parameter so that its value, when multiplied by the value of the DB_BLOCK_SIZE parameter, produces a number that is larger than the LVM stripe size.

Network options tuning for Transparent Application Failover

In case of a failover, UNIX clients may take up to 10 minutes to reconnect to the live instance; here's how to reduce this delay in the specific case of an AIX client.

The issue is a TCP Timeout and Retransmission problem, in that clients that are currently connected to the crashed node are not getting ACKs from the failed instance, so it keeps retransmitting the same data packet over and over again using an algorithm called Exponential Backoff (or exponential timeout/retransmit).

On AIX and other UNIX platforms that use Exponential Backoff, the default timeout value is 9 minutes (give or take a few seconds). This value is tunable via “no”, using the load time attributes rto_length, rto_low, and rto_high.

Using these three parameters, you can control *how often* and *how many times* to retransmit the same packet of data.

- How often is controlled by rto_low (the default is 1 sec.) and rto_high (the default is 64 sec.), which is used for retransmission based on the algorithm 1,3,6,12,24,48, and then 64 seconds apart (from here on).
- How many times a packet is retransmitted is controlled by rto_length (the default is 13).

Table D-1 lists these timeout and retransmission periods.

Table D-1 Timeout after 542 seconds (or approx 9.3 minutes)

Seconds	Retransmission
0	Initial transmission
1	1st
2	2nd
6	3rd
12	4th
24	5th
48	6th
64	7th
64	8th
64	9th
64	10th
64	11th
64	12th

Seconds	Retransmission
64	13th

If we reduce rto_length, or retransmission times, to 7, then the timeout is about 158 seconds (or approx. 2.5 minutes).

There are situations where you need to set rto_length to the max of 13. The most common reason for this is inferior network transmission quality, which can easily be tested via pinging (or by ftp'ing a big file) and measuring the amount of packet loss. If the packet loss is close to 20%, then 13 would be a good value for rto_length. However, if it is closer to 0%, then you can set rto_length to a value as low as 5.

You also have to consider factors such as Real-Time workload, the distance of the client from the server, and the amount of network traffic when tuning these rto_ parameters.

Checkpoints

A checkpoint occurs when the DBMS has to synchronize the data buffers with the log file. Keep in mind that many processes may be blocked when a checkpoint occurs. Frequent checkpoints write dirty buffers to the datafiles more often than otherwise. In a high update system, frequent checkpointing can reduce runtime performance.

Abbreviations and acronyms

AIX	Advanced Interactive Executive
CLVM	Concurrent Logical Volume Manager
CM	Connection Manager
DBWR	Database Writer
ESS	Enterprise Storage Server
GCS	Global Cache Service daemon
GES	Global Enqueue Service daemon
GPFS	General Parallel File System
GSD	Global Services Daemon
HACMP	High Availability Cluster Multi-Processing
HACMP/ES	High Availability Cluster Multi-Processing - Enhanced Scalability
HACMP/ESCRM	High Availability Cluster Multi-Processing - Enhanced Scalability Concurrent Resource Manager
HPS	High Performance Switch
IBM	International Business Machines Corporation
IPAT	IP Address Takeover
ITSO	International Technical Support Organization
JFS/JFS2	Journaled File System/2
LCK*n*	Lock process *n*
LMD	Lock Manager Daemon
LMS	Lock Manager Server
LUN	Logical Unit Number
LVM	Logical Volume Manager
NFS	Network File System
NPIC	Network Inter-Process Communicatiom
OUI	Oracle Universal Installer
PGA	Program Global Area
PMON	Program Monitor
PSSP	Parallel Systems Support Program
RAC	Real Application Clusters
RAM	Random Access Memory
RPD	RSCT Peer Domain
RSCT	Reliable Scalable Cluster Technology
SDD	Subsystem Device Driver
SGA	System Global Area
SMON	System Monitor
SPS	SP Switch
SPS2	SP Switch2
TAF	Transparent Application Failover
TCP/IP	Transmission Control Protocol - Internet Protocol
UDP	Universal Datagram Protocol
Vpath	Virtual Path
WWN	World Wide Number
WWPN	World Wide Port Number

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 242. Note that some of the documents referenced here may be available in softcopy only.

- *AIX 5L Differences Guide Version 5.2 Edition,* SG24-5765
- *Linux Clustering with CSM and GPFS,* SG24-6601
- *Implementing Oracle9i RAC with Linux on IBM xSeries eServers,* REDP0410

Other publications

These publications are also relevant as further information sources:

- *AIX 5L Version 5.2: Performance Management Guide*
- *IBM Reliable Scalable Cluster Technology for AIX 5L:Administration Guide,* SA22-7889
- *IBM Reliable Scalable Cluster Technology for AIX 5L: Messages,* SA22-7891
- *IBM Reliable Scalable Cluster Technology for AIX 5L:Technical Reference,* SA22-7890
- *RSCT Group Services Programming Guide and Reference,* SA22-7888
- *RSCT Event Management Programming Guide and Reference,* SA22-7354
- *RSCT First Failure Data Capture Programming Guide and Reference,* SA22-7454
- *General Parallel File System for AIX 5L:AIX Clusters Concepts,Planning,and Installation Guide,* GA22-7895
- *IBM General Parallel File for AIX 5L: AIX Clusters Administration and Programming Reference,* SA22-7900
- *Oracle9i Database Utilities Release 2 (9.2) Part No. A96652-01*
- *Oracle9i Real Application Clusters Administration Release 2(9.2)*
- *Oracle9i Installation Guide Release 2(9.2.0.1.0) for UNIX Systems: AIX-Based Systems, Compaq Tru64 UNIX, HP 9000 Series HP-UX, Linux Intel, and Sun Solaris*
- *Oracle9i Release Notes -Release 2(9.2.0.1.0) for AIX-Based 5L Systems (64-bit)*
- *Oracle9i Release Notes, Release 2 (9.2.0.1.0) for AIX-Based 5L Systems*
- *Configuring the IBM General Parallel File System (GPFS) in the Oracle Real Application Cluster (RAC) Environment,* by Rick Piasecki
- Oracle white paper, *“Optimal Storage Configuration Made Easy”,* by Juan Loaiza, Oracle Corporation
- *IBM’s General Parallel File System (GPFS) 1.4 for AIX, Architecture and Performance,* by Dominique Heger and Gautam Shah
- *IBM EAS Technical Brief, Configuring the Enterprise Storage Server (ESS) for Oracle OLTP Applications,* by Dale Martin

- *Quick Installation Guide Oracle9i RAC on IBM eServer pSeries with AIX 4.3.3,* by Fabienne Lepetit and Michel Passet
- *Configuring Oracle9i Real Application Clusters for High Availability on HACMP and AIX,* by Ronald M. Bautista
- *Step by Step Installation of RAC on IBM AIX (RS/6000),* Oracle Metalink white paper

Online resources

These Web sites and URLs are also relevant as further information sources:

- Oracle9*i* documentation

  `http://otn.oracle.com/docs/products/oracle9i/content.html`

- PTF download

  `https://techsupport.services.ibm.com/server/fixes`

- AIX ocumentation

  `http://www16.boulder.ibm.com/pseries/en_US/infocenter/base/aix.htm`

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

D

E

F

G

H

I

J

K

L

M

S

T

U

V

W

Z

Deploying Oracle 9*i* RAC on IBM @server Cluster 1600 with GPFS

(0.5” spine)
0.475”<->0.873”
250 <-> 459 pages

Deploying Oracle 9*i* RAC on IBM @server Cluster 1600 with GPFS

Redbooks

Oracle9i RAC cluster planning and installation on IBM pSeries running AIX

Availability and performance considerations

GPFS storage and networking considerations

This IBM Redbook explores the steps for installing Oracle 9*i* RAC in an IBM @server Cluster 1600. GPFS is implemented in the Cluster as the file system of choice for Oracle code and database files.

Performance and availability considerations are presented to show system architects, system administrators, and database administrators how to implement various features of a high performance, high availability database cluster environment.

This book also presents a set of high availability tests and scenarios to help in considering these aspects in the planning and implementation phases.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-6954-00 ISBN 0738499692