March 2011

Oracle Real Application Clusters on Linux on IBM System z: Set up and network performance tuning

Linux end to end Performance Team:
Dr. Juergen Doelle
Margaret Phillips

Table of Contents

About this publication

This is a study about installing and running Oracle Real Application Clusters (Oracle RAC) on Linux® on IBM System z®, with emphasis on performance related topics.

Authors

Linux end to end Performance team: Dr. Juergen Doelle, Margaret Phillips

Introduction

This study is about building and running Oracle RAC 10*g* Release 2 on Novell SUSE Linux Enterprise Server (SLES 10) SP2, with an IBM System z10®. The system setup is presented in detail. Several tests are conducted, varying various parameters, to determine optimal values and performance trade-offs.

Oracle Real Application Clusters (Oracle RAC) is a clustered variation of the Oracle relational database, called RDBMS. A Oracle RAC installation is made up of a set of clustered servers or nodes that share a single set of data stored on disk or other media and have an infrastructure to orchestrate the way that the nodes work together on the shared database to ensure data consistency.

Each *instance* or node of the cluster is a unique and fully operational database server. The shared data can be accessed by all the instances simultaneously, even at high levels of utilization. The technique that enables shared access to the stored data is called *cache fusion*. A service named Global Cache Service (GCS) maintains the cache on all the servers, and coordinates them to act as a single cache. The data is updated between the nodes with a fast interconnect device dedicated to that purpose.

This study is about installing and running Oracle RAC on Linux on IBM System z, with exploration of performance-related topics. The results of the study are based on experience obtained when building and using Oracle RAC as a test system in the IBM Linux Integration Test facility at IBM, specifically using Oracle RAC 10*g* Release 2 on Novell SUSE Linux Enterprise Server 10 (SLES 10) SP2 on an IBM System z10.

The goal of the study was to analyze the performance of an Oracle RAC environment running under Linux on IBM System z, with a transaction processing workload. The workload balanced mode was used with shared tables, in order to identify what factors influence cluster contention and how network tuning, especially on the interconnect, influences the performance. In this mode both nodes are actively used, the alternative is the failover mode where one node is used only in a failure situation.

This document is intended for people who are considering Oracle RAC as a clustered data base on Linux on IBM System z, and are interested in information derived from first hand experience. It is not presumed that the reader has previously installed or used Oracle RAC, or is familiar with the installation process or tuning choices.

This study focuses on features that are unique to IBM System z, including HiperSockets™, LPAR technology, and large disk storage units, so that the content can be useful to someone who is familiar with Oracle on other platforms and wants to know about the product on Linux on IBM System z.

This study describes the system setup, by stepping through the installation process that was followed, emphasizing features that are specific to IBM System z.

Summary

This paper is about installing and running Oracle RAC on Linux on IBM System z, with emphasis on performance related topics.

This study is about the Oracle RAC used in the workload balanced mode, which means that client requests are sent equally to both nodes. The resulting lock contention is intended for this test. The system setup is presented in detail. Several tests are conducted, varying various parameters, to determine optimal values and performance trade-offs with the focus on how the contention inside the cluster could be mitigated by the setup without changing the workload.

The amount of network traffic through the Interconnect between both nodes is an important indicator of the communication requirements needed to keep the caches consistent and coordinate access conflicts. The test setup run with a high cache hit ratio, which leads to very low disk I/O throughput values. This was important to ensure that the results are not influenced by disk I/O waits. The synchronization of the shared cache is managed on the database block level, meaning that even the smallest update statement causes a full block exchange over the interconnect when the data is accessed from both nodes. This exchange leads to much higher network traffic over the Interconnect than for client communications.

Comparing OCFS2 versus Oracle Automatic Storage Management (ASM) showed that ASM could be recommended as a cluster file system, providing better or at least comparable throughput at lower CPU utilization.

For an Oracle RAC in workload balanced mode the smallest possible block size is recommended, at least for tables that were updated from both nodes. Reducing the database block size from 8 KB to 4 KB leads, as expected, to a decrease in the Interconnect traffic of

approximately one-half. The transaction throughput increased by 4%, while the cluster wait events are reduced by 4%. This block size reduction is also correlated with reduced lock contention. For example, locks for two different 4 KB blocks in the same 8 KB block are now two independent locks, making the sending of the block unnecessary.

An important tuning improvement for the test environment was the change from shared server processes to dedicated server processes. With that change, the same system was able to drive so many more transactions that adding more CPU capacity leads finally to an improvement of a factor of three. This effect contributed to our workload with a moderate amount of clients (up to 80), where each client is 100% active, generating a continuous workload, which easily keeps one Oracle server process busy. If several of these workload generating clients are sharing a server process, this server process becomes a bottleneck. For this type of workload pattern, the use of dedicated server processes is recommended. This workload pattern is typically produced by a batch job, an application server, or a connection concentrator.

The opposite scenario occurs with connections that have a low utilization, such as from any manual interactions, and with a very high number (1000 or more) of users. Here, the use of dedicated server processes results in thousands of under-utilized server processes, with their corresponding memory requirements. In the test case, the memory usage increases when scaling the number of users up to 40, then the total memory usage leveled off.

When scaling the number of users, the behavior of the system is determined by two types of lock contentions: globally between the two nodes or locally inside each node. The local lock contention becomes the dominant factor when the number of users increases. The lock contention is a bottleneck that needs to be resolved if user activity is to be increased. In our case, the cause of lock contention is updates being made to a small shared table. Lock contention occurs either when the nodes in the cluster, or two users inside the nodes, are trying to update the same data. The local contention is based on row locking, while the cluster contention locks the whole data block.

For tuning the network setup, it seems that the typical network tuning recommendations should be carefully applied to Oracle RAC running in a workload balanced mode. Increasing the network throughput might be related to increasing the cluster contention, which could lead to a performance degradation. For example, increasing the buffer counts or the MTU sizes did not improve the throughput in the test scenarios. The best results were obtained with the buffer count at its default (16 buffers). For the MTU size, it seems that Oracle RAC and the Oracle client use a maximum package size of 1330 bytes for all TCP/IP connections, therefore the recommendation is to use an MTU size of 1492 for the LAN connection and 8192 for the Interconnect.

The device queue length was found to be a very important tuning parameter. The increase to 2000 (default 1000) caused an improvement of approximately 40%, which is a very high value. But the observation that there is a size that is too large, resulting in degradation, indicates the need for careful use of this parameter. Increasing this parameter is best done in combination with monitoring of the network or transaction throughput.

For the Interconnect network device type, HiperSockets are the best choice for the Oracle RAC interconnect, when the server nodes are on the same physical IBM System z machine. Because it is not recommended to run all Oracle RAC nodes on one physical machine, the second choice would be to use a 10 Gb OSA card. The use of a 10 Gb OSA connection does not necessarily require a full 10 Gb infrastructure with switches and so forth. The test scenarios used a direct coupling of two OSA ports with a standard cable. This provides a connection in point-to-point mode, which additionally reserves the full bandwidth for the interconnect and avoids any interaction with other network workloads, ensuring a constant response time. With a 1 Gb OSA connection for the Interconnect, a new wait event occurred, wait event 'cr request retry', indicating that the 1 Gb OSA interconnect is not sufficient for the workload processed.

The new IBM zEnterprise™ 196 (z196) provides an impressive improvement of the workload driven per CPU of 45%, which makes it very attractive for this workload. These results are comparable with other Linux on IBM zEnterprise 196 results, for example shown at: http://www.ibm.com/developerworks/linux/linux390/perf/tuning_z10.html#z196

Be aware that the contention inside the cluster is one important parameter that limits additional improvements due solely to faster processor speed.

Hardware environment

For the Oracle Real Application Clusters study, IBM System z hardware and various softwarethat runs on Linux were employed.

IBM System z hardware used in the Oracle Real Application Clusters study

This section lists the hardware used in the Oracle RAC study.

Table 1 lists the host hardware for each node in the Oracle RAC cluster.

Table 1. Host hardware

System	Operating System	Number of CPUs	Memory in GB	Network	Architecture
Oracle RAC node1	SLES 10 SP2	Two or three, dedicated	16	▪ One 1 Gb OSA ▪ One HiperSockets or 10 Gb OSA	z10™ LPAR
Oracle RAC node2	SLES 10 SP2	Two or three, dedicated	16	▪ One 1 Gb OSA ▪ One HiperSockets or 10 Gb OSA	z10 LPAR

The IBM System z10 had these features.

- Two LPARs, each with
  - Two or three processors, dedicated
  - 16 GB of memory
  - HiperSockets connection
  - One OSA Express3 1 Gb Ethernet
  - One OSA Express3 10 Gb Ethernet
- Eight FICON® Express 4 (4 GB), shared
- ECKD™ disk devices (DASD) on IBM System Storage® DS8000®
  - 16 ranks are spread across the logical configuration of the control unit to avoid congestion.
  - The storage server is connected to the IBM System z with eight FICON paths.

Client hardware

The client system was an IBM System x® running Linux with the Oracle client software. This client system was used as a workload generator for the OLTP workload.

Table 2 provides details about the client system.

Table 2. Client hardware used in the Oracle RAC study

System	Operating System	Number of CPUs	Memory in GB	Network	Architecture
Workload generator	Red Hat Enterprise Linux 4.3	Four	2	1 Gb OSA	IBM System x

Network setup

This section lists the network setup of the Oracle Real Application Clusters study.

The network setup uses these features:

- For the private Interconnect between the Oracle RAC nodes.
  - HiperSockets
  - 10 Gb Ethernet
  - 1 Gb Ethernet
- For the LAN connection to the workload generating client
  - 1 Gb Ethernet
- Oracle RAC requirement for external range IP address for public IP addresses
- Networking Switch with VLANs and a private LAN, in order to isolate I/O traffic

Each Oracle RAC node uses three IP Addresses and host names on two separated network interfaces, as described in Table 3.

Table 3. IP addresses and host names for the network interface

IP interface	Function	Network interface	Network hardware
db-node*n*	External Ethernet access	eth0	One OSA Express 1 Gb Ethernet
db-node*n*vip	Access for clients	eth0:1 alias to eth0	
db-node*n*priv	Interconnect	hsi0	HiperSockets or OSA Express 10 Gb Ethernet
Note: *n* = 1, 2, ... corresponds to the node number			

Figure1 illustrates the network connection.

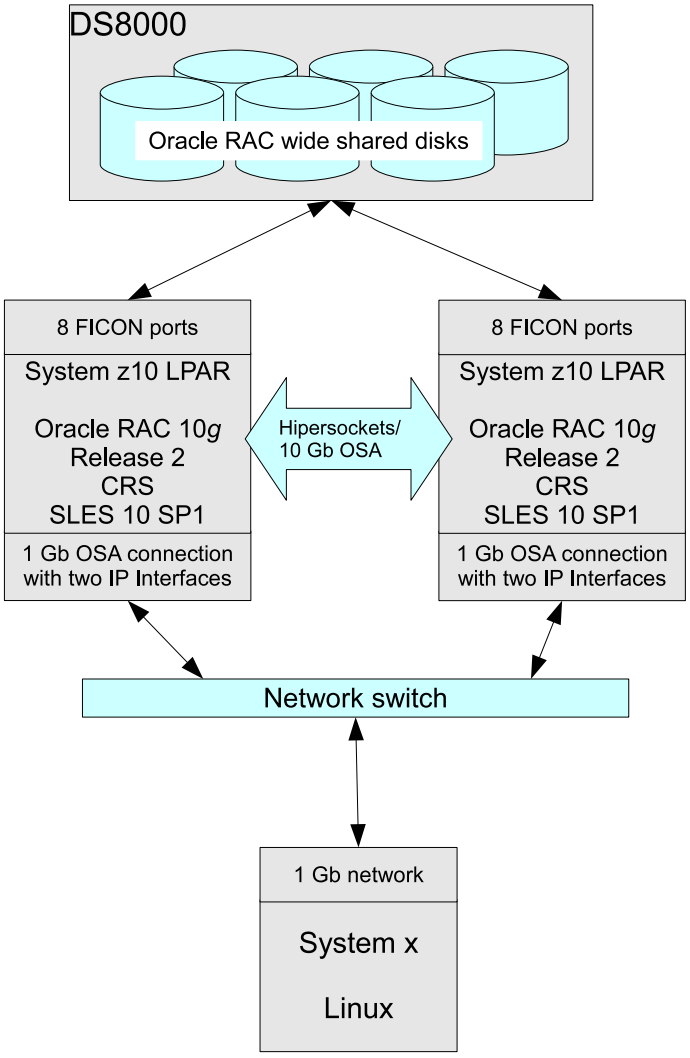

Figure 1. Network connections used in the Oracle RAC study

HiperSockets

HiperSockets are a high speed and high-bandwidth I/O function on IBM System z that communicates between Logical Partitions (LPARs) inside a central processor complex (CPC). HiperSockets are configured as a standard Linux network device, providing point to point connectivity.

10 Gb OSA setup

When the 10 Gb OSA card was used as the interconnect for the Oracle Cache Fusion function, it was connected from one OSA port to the other with a cable attached directly between the two channel adapters, and without a switch in the middle. This setup enables the use of a 10 Gb network environment as a point to point connection without the requirement of a 10 Gb LAN infrastructure. The use of both ports (instead of a shared port) closely resembles a configuration of a 10 Gb cabled connection between two central processor complexes (CPCs).

1 Gb OSA setup

For this setup the standard LAN infrastructure was used.

Software used in the Oracle Real Application Clusters study

This section lists the software used in the Oracle RAC study.

Table 4 lists the software used in the Oracle RAC study.

Table 4. Software used in the Oracle RAC study

Product	Version and Release	Comments
Oracle RAC 10*g* Release 2, including: ▪ Oracle Clusterware ▪ Oracle RDBMS ▪ Oracle Automatic Storage Manager (ASM)	10.2.0.2 base installation (CRS, RDBMS, ASM) 10.2.0.4 patch p6810189 (CRS, RDBMS, ASM)	
Novell SUSE	SLES 10 SP2	Oracle RAC System z installed in LPARs
Red Hat Enterprise Linux AS	Release 4 (Nahant Update 3)	For the x86 client

Oracle client system setup

The client system was an IBM System x running Linux with the Oracle client software. This client system was used as a workload generator for the OLTP workload. The software for the client system was downloaded from the Oracle Technical Network (OTN), along with the download of Oracle RAC 10*g* Release 2 version 10.2.0.4. For details about how to download software from OTN, see Planning to build an Oracle Real Application Clusters system.

About Oracle Real Application Clusters on Linux on IBM System z

This study focuses on Oracle RAC clusters using Linux on IBM System z processors. A brief history or Oracle RAC on IBM System z is presented, followed by specific information about Oracle RAC components, voting disks, and modes of operation.

History of Oracle RAC

Oracle relational databases have been available for IBM platforms for many years, and they are part of many solutions on enterprise IBM servers. The history of Oracle relational databases on IBM servers began with IBM System p®, and was then introduced to z/OS® on IBM System z.

When IBM System z became a Linux platform, there were early development versions of Oracle RDBMS with Oracle Real Application Clusters for 31-bit Linux. Oracle RAC 10*g* Release 2 for Linux on IBM eServer™ zSeries® was introduced in 2008 with version 10.2.0.2, which is not an Oracle certified release.

Version 10.2.0.2 was superseded with version 10.2.0.3, which is Oracle certified. However 10.2.0.3 became upgradeable to 10.2.0.4, also Oracle certified. The Oracle product that is used in this study is based on Oracle RAC 10*g* Release 2 version 10.2.0.4 for Linux on System z. At the time of this writing, 10.2.0.2 must be installed as a base level, and then patched to the higher levels of 10.2.0.3 or 10.2.0.4.

Reasons to use Oracle RAC on IBM System z

The IBM System z environment is one that enterprise users go to as the platform for reliability, availability, and large scalability. These are the same enterprise requirements that Oracle RAC serves. As enterprise environments evolve, the successful tools and platforms make strategic jumps across boundaries to find solutions to the latest types of problems.

Oracle RAC provides the efficiency of having a single source for real-time data that can be accessed from many applications. For example, multiple government agencies could share a database that is being updated with different types of information, rather than maintaining multiple databases and having to do comparisons of redundant data.

Oracle RAC on IBM System z can provide the economy of not maintaining multiple databases. The size of some databases is in terabytes or petabytes. This size makes keeping multiple copies, in terms of disk storage, very expensive. With the high security of IBM disk storage systems, backup and recovery can be maintained and costs can be controlled.

The importance of failover protection is a primary driver in the combination of Linux on IBM System z and Oracle RAC. Data is protected in the event of failure of the database instance, not the data on disk. Oracle RAC maintains change logs, and in the event of a failure or interruption of service no data will be lost. Oracle RAC can be an important aspect of a High Availability plan in the Linux on IBM System z environment.

One of the goals in enterprise computing today is consolidation of distributed workloads onto Linux on IBM System z in order to take advantage of virtualization. The availability of Oracle RAC on IBM System z makes consolidation onto the larger mainframes attractive for users who would not have otherwise been able to migrate a clustered database.

For Oracle RAC users who want multiple instances in LPARs on IBM System z, HiperSockets is a unique and fast connection that is well suited for the Oracle RAC interconnect, and is only available on IBM System z.

Oracle has a large offering of middleware business products and is constantly bringing more of them into certifiable status on Linux on IBM System z. Recent announcements were for Oracle GoldenGate, Oracle Fusion Applications, Weblogic Server, E-Business Suite, and Peoplesoft Tools. Overall, this offering is a comprehensive stack based around Oracle Database 10*g* Release 2.

Components of Oracle Real Application Clusters

The three main components of Oracle Real Application Clusters are described here.

Oracle RAC consist of three components:

- The Oracle clusterware software (CRS)
  - Software to create and manages the communication between the cluster of nodes
- The relational database (RDBMS)
  - The same relational database that can be installed as a stand-alone database

- The Automatic Storage Manager (ASM)
  - A disk manager program that provides the capability to share data files stored on disk arrays.

Figure 2 illustrates these three components and their responsibilities.

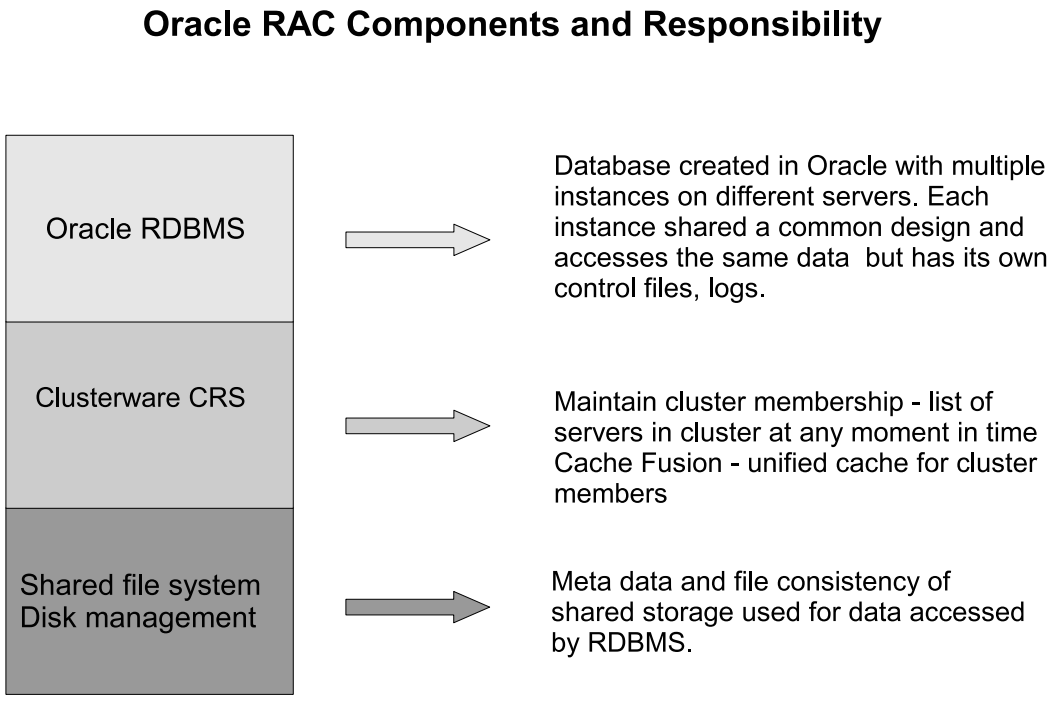

Figure 2. Oracle RAC components

The Oracle Cluster Registry and the Voting Disk

There are two storage locations of Oracle Real Application Clusters. They are the Oracle Cluster Registry and the Voting Disk.

The Oracle RAC uses two storage locations, which can be accessed by all cluster members:

- The Oracle Cluster Registry (OCR)
  - Contains information for managing the cluster
- The Voting Disk
  - Contains information to determine which nodes are active members of the cluster at any given moment

In Oracle RAC 10*g* Release 2, these storage locations can be files in a shared file format such as OCFS2, or they can be Linux configured raw storage or block Linux devices. A block device in Linux is an unformatted partitioned disk that might be named, for example, `/dev/dasda1`, and shared by bringing the device online on all of the nodes in the cluster. It is not necessary for either the OCR or Voting Disk to be full disks.

Although block devices are not configured as raw storage, Oracle has direct access and uses them as raw storage.

Devices configured in Linux as raw devices are being deprecated, but depending on the Linux distribution they might be used in Oracle RAC 10.2.0.2 and 10.2.0.3. In the case of using raw devices for later versions, consult Oracle documentation for instructions.

The following instructions help to prepare block devices for use as OCR and Voting Disk and backups. After these components are set up, when clusterware is installed nothing more must be done with them by the administrator, unless the devices must be replaced for some reason.

Each OCR or Voting Disk and each back-up must have 100 MB of storage available.

1. Create a partition table for the DASD by issuing this command:

   ```
   fsdasd -a /dev/dasdn
   ```

2. Clear the residual data with the following command:

   ```
   dd  if=/dev/zero of=/dev/dasdn1 bs=1M count=2347
   ```

3. Change the disk owner to the user **oracle** for the Voting Disk and back ups.
4. For all the disks, change the group to **oinstall**.
5. For all the disks, set the MODE to 0660.

   Note: After the installation, the owner of OCR is to the user **oracle** automatically.

6. In Linux SLES 10 SP2, create a file in `/etc/udev/rules.d` named `98-oracle.permissions.rules` with these lines:

```
# for partitions import parent information
KERNEL=="*[0-9]", IMPORT{parent}=="ID_*"
# OCR disks
KERNEL=="dasdf1", OWNER="root", GROUP="oinstall" MODE="0660"
KERNEL=="dasdp1", OWNER="root", GROUP="oinstall" MODE="0660"
# VOTING DISKS
KERNEL=="dasdg1", OWNER="oracle", GROUP="oinstall" MODE="0660"
KERNEL=="dasdq1", OWNER="oracle", GROUP="oinstall" MODE="0660"
```

This file shows two block devices for OCR and a back-up, and two more for Voting Disks.

IMPORTANT

It is highly recommended to maintain cluster-wide persistent disk device names. For details, see Setting up persistent names for disk devices

Oracle Real Application Clusters operation modes

Oracle Real Application Clusters has two modes of operation: Fail-Over and Workload balance.

Fail-Over

Only one node processes client requests. For high availability, the second node (called the *failover node*) is ready in a standby mode, and is activated when the first node fails. Cache consistency between both nodes is still required, to ensure that in the event of an error, the failover node works on current data.

Workload balance

Multiple nodes are processing client requests. This processing requires, in addition to the cache consistency, a locking mechanism between the nodes to ensure update consistency.

In case of failover, the IP address with the vip suffix is assigned to the network device of the failover node.

Planning

To plan the installation used in this study, it is necessary to understand some basic Oracle terminology, the installation of various Oracle versions, and the mandatory installation sequence.

Planning to set up Oracle Real Application Clusters on IBM System z

There are several major decision points when planning to build an Oracle Real Application Clusters system on Linux on IBM System z, such as:

- Disk storage type (DASD or SCSI)
- Network device type for the Oracle Interconnect - HiperSockets, 10 Gb Ethernet, or 1 Gb Ethernet
- Linux distribution
- Shared disk management (Cluster file system or ASM)

Disk storage type

With IBM System z, the storage can be set up as disk storage type DASD or SCSI. This study used the device type DASD, which implements the Extended Count Key Data (ECKD) interface. The FICON Express cards that are the channel adapters are set up as type FICON. The IBM developerWorks® website provides techniques for working with disk devices on IBM System z with Linux. See http://www.ibm.com/developerworks/linux/linux390/perf/tuning_diskio.html

Network device type for the Oracle interconnect

HiperSockets

HiperSockets are a high speed and high-bandwidth I/O function on IBM System z that communicates between Logical Partitions (LPARs). While the interface for a HiperSockets connection is configured on Linux on IBM System z in the same manner as an OSA device, the transmission of the data takes place within the central processor complex (CPC) using the memory bus rather than the I/O bus, and this data transmission has a very short latency time. In an ordinary transmission on IBM System z, every I/O must minimally be processed by the I/O subsystem, including the channel adapter, and then connected externally.

HiperSockets is also a secure way to handle data traffic, because it is an isolated point to point connection. Oracle RAC clustering requires a fast private interconnect, to keep the server nodes' shared caches consistent. HiperSockets hardware should be the first choice for Oracle RAC nodes on LPARs on the same physical IBM System z.

1 Gb versus 10 Gb Ethernet

An alternative interconnect device is the 10 Gb OSA card. It has a bandwidth of 10 Gb. The use of a 1 Gb network connection for the Oracle RAC interconnect should be considered very carefully.

Linux distribution on IBM System z

Both Linux distributions, Novell SUSE Linux Enterprise Server (SLES) and Red Hat Enterprise Linux are supported for Oracle RAC on Linux on IBM System z. This study used SLES 10 SP2.

Shared disk management

When planning to manage shared disks, a decision must be made between using OCFS2 and the Oracle Automatic Storage Manager.

OCFS2 Cluster file system versus ASM

To share data in a cluster, it is essential to have a file system that is able to coordinate accesses from independent nodes to shared files, while ensuring data consistency. There are two cluster file systems supported for Oracle RAC for Linux on IBM System z:

- OCFS2
- Oracle Automatic Storage Manager (ASM), which is distributed with the Oracle RDBMS package.

ASM is a special solution for Oracle databases and cannot be used as a standard file system.

ASM is an application that manages every aspect of storage of data on a set of disks, from the lowest level where it determines where the bits are physically stored, providing a virtual file system that looks and acts like a hierarchical file system. The disks that are under management by ASM are not accessible to Linux as other mounted file systems, but these disks are available as virtualized storage that can be managed through the user interface program.

Note: It is not recommended or required to stack the volume managers ASM and LVM. Because ASM stripes the data over the disks automatically, the usage of the Linux logical volume manager (LVM) is not required for striping. On the current distributions, multi-pathing with FCP disks is handled by the multipath daemon.

From the user perspective, a file system controlled by ASM is similar to a normal file system, although it is accessed using the command asmcmd in the ASM HOME.

An alternative to ASM is to use OCFS2, one of the Linux file systems for shared storage. To the user, a file controlled by OCFS2 appears as a normal file, but OCFS2 supports shared access to files without corruption of the data. For more information about OCFS2 and how it can be used on IBM System z, refer to this document on DeveloperWorks: http://www.ibm.com/developerworks/linux/linux390/perf/tuning_filesystems.html

OCFS2 Version 1.4 comes as part of the SLES 10 SP2 distribution. OCFS2 Version 1.2 comes as part of the SLES 10 SP1 distribution. OCFS2 exists on the SLES 11 distribution in the HA Extension.

The most serious shortcoming for OCFS2 on large databases is that OCFS2 does not do data striping like ASM does. The administrator would have to arrange to do striping of data on the disks using the control program on the IBM System Storage DS8000 disk array (storage pool striping), before configuring the storage for the Oracle RAC system.

The setup of OCFS2 is described in this white paper: http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/index.jsp?topic=/liaag/l0wocf00_2010.htm

Oracle Automatic Storage Manager

The disks that are managed by the Oracle Automatic Storage Manager (ASM) in Oracle RAC 10.2.0.2 and 10.2.0.3 can be configured as raw devices. With ASM in 10.2.0.3 and 10.2.0.4, block devices are also supported as storage that is easier to set up and produces the same results. The Linux support for raw devices is deprecated and might be removed, which is discussed in The Oracle Cluster Registry and the Voting Disk.

ASM uses direct I/O and is able to provide striping of the disk space. ASM manages the disks that are assigned to it down to the logic that operates on the tracks on the disks. To the system administrator ASM provides a virtualized file system. The file system can be accessed by the user named **oracle** using the command asmcmd from the Oracle $HOME_ASM environment. Storage is logically organized around *diskgroups* created by the user. Diskgroups can easily be created and have disks added or removed from them. ASM uses direct I/O.

In planning for ASM, select the disks to use for data files that are spread across the available disk arrays according to the model. To understand the logical structures in the IBM System Storage DS8000 storage server, consult IBM Redbooks® publication *DS8000 Performance Monitoring and Tuning*, which is linked in Appendix. References.

Planning to build an Oracle Real Application Clusters system

When planning to build an Oracle RAC cluster, you must first choose an Oracle RAC version, download it and then set up the home directories properly.

Versions of Oracle RAC

Oracle RAC 10*g* Release 2 version 10.2.0.2 is not certified by Oracle for Linux on IBM System z. However, versions 10.2.0.3 and 10.2.0.4 are certified by Oracle for Linux on IBM System z. To install versions 10.2.0.3 or 10.2.0.4, it is necessary to begin by installing version 10.2.0.2 and then apply the upgrade as a patch.

The download of Oracle Database 10*g* Release 2 (10.2.0.2) for Linux on IBM System z is available from the Oracle Technical Network (OTN) at:
http://www.oracle.com/technetwork/database/enterprise-edition/downloads/index.html

Complete these steps to perform the download from this Web site:

1. In the **Downloads** tab, select **11g Enterprise/Standard Editions**.
2. Scroll down to locate the **10g Release2 Standard and Enterprise Edition** section.
3. Make a selection for Linux running on IBM System z (z/Linux).
4. Read and accept the License Agreement.
5. Select **Download the Complete Files**.
6. On the **Download** page, select these two files:
   - `10202_zlinux_database.zip`
   - `10202_zlinux_clusterware.zip`

The database package also contains ASM.

For the upgrade to version 10.2.0.4, download patchset `p6810189_10204_LINUX-zSer/zip` from a different location: Oracle's "My Oracle Support" formerly called Metalink. The patchset is named **10.2.0.4 Patch Set for Oracle Database Server for platform IBM: Linux on System z**, and it contains clusterware and ASM as well as RDBMS, the database.

Oracle releases bundled patches (called CPUs - Quarterly Critical Patch Updates) on a regular schedule. They are roll up patches, which incorporate the prior updates to the release. For example, the fourth CPU to Oracle Real Application Clusters for Linux on IBM System z takes Oracle RAC version 10.2.0.4.1 to version 10.2.0.4.4. CPUs for Linux on IBM System z can be applied with OPatch, a patch applicator that is part of Oracle RAC 10*g* Release 2.

When searching OTN and other Oracle sources for products applicable to IBM System z, it may be necessary to use search strings that are not the current IBM names, such as **zSeries**, **Linux on System z** or **S/390®**.

Note: Although you can download Oracle RAC and its patches and associated software at any time, do not install these products until after you have properly prepared the target Linux systems. See Preparing Linux for the installation of Oracle Real Application Clusters.

Oracle HOME directories and multiple Oracle databases

The test system was built with all the Oracle components (ASM, CRS, and RDBMS) at the same release level. However, the Oracle RAC model can be built with additional databases at different release levels on the same server, as long as the clusterware (CRS) on any server is at the same or a later release level than the level of any of the other components, such as RDBMS or ASM. The additional databases can be single instance or in a cluster. Each database can be upgraded individually, while support continues for databases at an earlier level.

Operating in an Oracle environment with many possibilities is accomplished by maintaining sets of environmental variables such as $HOME, which would have a related $SID to identify a particular database, and $PATH, $LIBPATH and other environmental variables as needed. As an example, $HOME_CRS is set to `/product/crs`, which is the location in this installation of the Oracle RAC 10.2.0.4 CRS binaries, CRS libraries, CRS logs, and CRS parameters.

The database administrator's files, such as scripts and parameters for creating and maintaining databases, is located outside of the Oracle HOMEs. The actual data is located in the shared storage environment such as ASM.

In an Oracle clustered environment, each database can be upgraded and expanded individually while support continues for databases at earlier releases.

Oracle server processes - shared versus dedicated

After the clustered database is installed, one of the decisions to make is whether to use *dedicated server processes* or *shared server processes*. Dedicated server processes choose a one to one relationship between request and server. With the other choice, shared server processes, the requests are queued to match processes from any of the servers being served by dispatchers.

The efficiency of each approach is related to the type of activity that takes place. With the dedicated server setup, a client request to the database connects to a server process and holds

it until the request is completed. In this study, this process provided very good throughput results. This configuration might use a large amount of memory with an increasing amount of client connections, because each server process requires a certain amount of memory. The shared server process reuses the same server process for multiple client processes, which goes the opposite way, reduced memory usage with possibly lower throughput.

When the workload is determined by many (1000 or more) connected users who spend significant time between actions, such as online shoppers, the shared server process setup might be more efficient. For workloads with a concentrated high utilization on each connection, for example when the Oracle Database is the backend from an application server or behind a connection concentrator, dedicated server processes might be more appropriate.

Details on how to configure shared or dedicated servers are in the section Setting up the listeners.

Installation and configuration

The installation and configuration of an Oracle Real Application Clusters system entails first preparing the target Linux systems, and then performing the installation tasks.

Preparing Linux for the installation of Oracle Real Application Clusters

Linux preparation before the installation of Oracle Real Application Clusters consists of preparing users, specifying various parameters and settings, setting up the network, setting up the file system, and performing some disk tasks.

To prepare Linux for Oracle RAC installation, complete these tasks in the order that they are listed.

Creating users and authentication parameters

To set up users and authentication for Oracle RAC, complete the following steps:

1. Log in with root authority.
2. Create the user named **oracle** on all of the nodes.
3. Create the user group named **oinstall** on all of the nodes.
4. Use an editor such as vi to add these lines to the body of the `/etc/security/limits/conf` file, to increase the limits for open files and processes:

```
oracle          hard    nofile         65536
oracle          soft    nproc           2047
oracle          hard    nproc          16384
# End of file
```

5. Check these four files for the following line, and add it if the line is not already present:

```
session   required   pam_limits.so
/etc/pam.d sshd
/etc/pam.d/login
/etc/pam.d/su
/etc/pam.d/xdm
```

Using ulimit for shell settings

Set shell limits by editing the file with suffix `.profile` for the user oracle. Insert the following lines in the file:

```
ulimit -n 65536
ulimit -u 16384
export OPATCH_PLATFORM_ID=211
```

The environment variable OPATCH_PLATFORM_ID indicates Linux on System z.

Setting Linux kernel parameters

Check the Linux kernel parameters in the `/etc/sysctl.conf` file. The values of some of the parameters must be increased if they are not equal to or greater than these values:

```
kernel.sem = 250 32000 100 128
fs.file-max = 65536
net.ipv4.iiiip_local_port_range = 1024 65000
net.core.rmem_default = 1048576
net.core.rmem_max = 1048576
net.core.wmem_default = 262144
net.core.wmem_max = 262144
```

Setting up ssh user equivalence

Oracle uses the ssh protocol for issuing remote commands. Oracle uses scp to perform the installation. Therefore, it is necessary for the users **oracle** and **root** to have *user equivalence*, or the ability to use ssh to move from one node to another without authenticating with passwords or passphrases. Set up ssh user equivalence between each of the interfaces on all of the nodes in the cluster with public and private authentication key pairs that are generated with the Linux command key-gen. Instructions on how to use key-gen to create public and private security keys are available at:
http://www.novell.com/documentation/sles10/sles_admin/?page=/documentation/sles10/sles_admin/data/sec_ssh_authentic.html

When the key-gen command issues a prompt to enter a passphrase, just type enter so that no passphrase will be required.

For example, an installation with two Oracle RAC servers will have a total of twenty-four key pairs (between three interfaces, public, vip, and interconnect; users root and oracle; node1 to node; node1 to node2; and the reverse).

Note: After the user equivalence has been set up, try to use each of the new authorizations one time before proceeding with the installation. This step is necessary because the first time that the new authorizations are used, they generate an interactive message that the Oracle Universal Installer cannot properly handle.

Setting up the network for Oracle RAC

Each server node in the cluster needs two physical connections and three IP interfaces, which must be set up before beginning the installation. Network device type for the Oracle interconnect explained the need to use the device that is fastest and can handle the most throughput and traffic for the private interconnect between server nodes in the cluster. For this reason, the study used HiperSockets on IBM System z.

For the public access, a 1 Gb Ethernet was used on each server, which was configured to have two interfaces, the second one being an alias.

For example, on the first server (node1) the interface for a particular OSA device port was set up as an Ethernet connection with a hardware definition in `/etc/sysconfig/network`, where a file named `ifcfg-qeth-bus-ccw-0.0.07c0` where 07C0 is the device address of an OSA port contains the following lines:

```
NAME='IBM OSA Express Network card (0.0.07c0)'
IPADDR='10.10.10.200'
NETMASK='255.255.255.0'
```

By using the network address 10.10.10.200 and a netmask of 255.255.255.0, it leaves the hardware device available to be used by any other interface using the network address of the form: 10.10.10.xxx, and the clusterware startup scripts will create an alias for the public interface when the node starts CRS.

This environment does not use a DNS server, so the alias interface name must be included in the file `/etc/hosts`, where the two host addresses and names looked like this:

```
ff02::3         ipv6-allhosts
10.10.10.200    rac-node1 rac-node1.pdl.pok.ibm.com
10.10.10.202    rac-node1vip rac-node1vip.pdl.pok.ibm.com
```

In an Oracle RAC system, the string **VIP** in the interface name stands for virtual IP, and identifies its role. Having two interfaces for the same Ethernet connection supports immediate failover within the cluster. If a node is not responding, the vip IP address is attached to another node in the cluster, faster than the time it would take for hardware timeout to be recognized and processed.

The larger section of the file `/etc/hosts` shows how the aliases were used for the two nodes, because they are all present in the file and it is also the same on the other node:

```
10.10.10.200    db-node1 db-node1.pdl.pok.ibm.com
10.10.10.201    db-node2 db-node2.pdl.pok.ibm.com
10.10.10.202    db-node1vip db-node1vip.pdl.pok.ibm.com
10.10.10.203    db-node2vip db-node2vip.pdl.pok.ibm.com
10.10.50.200    db-node1priv db-node1priv.pdl.pok.ibm.com
10.10.50.201    db-node2priv db-node2priv.pdl.pok.ibm.com
```

The Linux command ifconfig displays interface net0 and its alias named net0:1

```
net0   Link encap:Ethernet  HWaddr 00:14:5E:78:1D:14
       inet addr:10.10.10.200  Bcast:10.10.10.255  Mask:255.255.255.0
       inet6 addr: fe80::14:5e00:578:1d14/64 Scope:Link
       UP BROADCAST RUNNING MULTICAST  MTU:1492  Metric:1
        RX packets:12723074 errors:0 dropped:0 overruns:0 frame:0
       TX packets:13030111 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:1000
       RX bytes:2164181117 (2063.9 Mb)  TX bytes:5136519940 (4898.5 Mb)

net0:1 Link encap:Ethernet  HWaddr 00:14:5E:78:1D:14
        net addr:10.10.10.203  Bcast:10.10.10.255  Mask:255.255.255.0
       UP BROADCAST RUNNING MULTICAST  MTU:1492  Metric:1
```

When Oracle RAC uses the public interface, it will select the interface with the number zero after the prefix. By default this interface would be **eth0**. If there is a situation where the public interface must be forced to use number 0, use the udev command to name the interfaces. Naming the interfaces can be done by modifying or creating the `/etc/udev/rules.d/30-net_persistent_names.rules` file, to contain a line like this example, where 07C0 is the device and **net** is a prefix that was chosen in order to be different from **eth**

```
SUBSYSTEM=="net", ACTION=="add", ENV{PHYSDEVPATH}=="*0.0.07c0",
IMPORT="/lib/udev/rename_netiface %k net0"
```

A separate physical connection will become the private interconnect used for Oracle *cache fusion*, where the nodes in the cluster exchange cache memory and messages in order to maintain a united cache for the cluster. This study used IBM System z HiperSockets in the first installation, and named the interface db-node1priv, since the Oracle convention is to call the interconnect the private connection.

The decision of what type of connectivity to use for the private interconnect is an important decision for a new installation, the objective being high speed, and the ability to handle transferring large amounts of data.

The example of the interconnect for both nodes is shown in the example above of `/etc/hosts`.

The first node (node1) uses IP address 10.10.50.200 and host name db-node1priv. The second node (node2) uses IP address 10.10.50.201 and host name db-node1priv.

Using a 10.x.x.x network for the external (public) interfaces

The setup for the study required circumventing an Oracle RAC requirement that the external RAC IP address and it's alias be an IP address that is from the range of public IP addresses. Th setup needed to use an IP address in the form 10.10.x.x, which is classified as an internal range.

Oracle RAC would not work until a change was made in `$HOME_CRS/bin/racgvip`. To do the same thing on your system, either set the variable DEFAULTGW to an IP address that can always be pinged successfully (it will never be used, Oracle only checks to see if it is available), or else search for and change FAIL_WHEN_DEFAULTGW_NOT_FOUND to a value of 0.

Using udev when preparing to install Automatic Storage Manager

To set up shared storage for Oracle RAC on IBM System z, some of the attributes in Linux of the DASD disk storage devices must be modified. In SLES, configuration files located in `/etc/udev/rules.d` are read by Linux shortly after the kernel is loaded, and before Linux has created the file structures for disk storage and also before Linux assigns file names and attributes to all the known disk devices and partitions.

The udev command can be used to change ownership of the block devices that are used for the OCR and Voting Disks. It is also necessary to alter the attributes of the block devices that will be given to ASM to manage as shared storage for data. Shared DASD used for data also managed by ASM must be assigned the owner **oracle** and the group named **dba**.

For this study, a new file was created and given a high number (98) in the file name, so that it would be read last and the setup changes would not be overwritten by other startup processes. There are different rules for the udev command even when comparing SLES 10 SP2 with SLES 10 SP1, so it may be necessary to check the man pages or documentation for the udev command on your system, to ensure that it works as expected.

This is a sample for the udev file `98-oracle.permissions.rules`:

```
# for partitions import parent information
KERNEL=="*[0-9]", IMPORT{parent}=="ID_*"
# OCR disks
KERNEL=="dasdf1", OWNER="oracle", GROUP="oinstall" MODE="0660"
KERNEL=="dasdp1", OWNER="oracle", GROUP="oinstall" MODE="0660"
# VOTING DISKS
KERNEL=="dasdg1", OWNER="oracle", GROUP="oinstall" MODE="0660"
KERNEL=="dasdq1", OWNER="oracle", GROUP="oinstall" MODE="0660"
#ASM
KERNEL=="dasdh1", OWNER="oracle", GROUP="dba" MODE="0660"
KERNEL=="dasdi1", OWNER="oracle", GROUP="dba" MODE="0660"
KERNEL=="dasdj1", OWNER="oracle", GROUP="dba" MODE="0660"
KERNEL=="dasdk1", OWNER="oracle", GROUP="dba" MODE="0660"
KERNEL=="dasdm1", OWNER="oracle", GROUP="dba" MODE="0660"
KERNEL=="dasdn1", OWNER="oracle", GROUP="dba" MODE="0660"
```

To make the changes take effect immediately, run this command:

```
/etc/init.d/boot.udev restart
```

Setting up persistent names for disk devices

Linux assigns names to all devices that it discovers at startup, in the order in which it discovers them, assigning the device names starting with the name dasda (or sda for SCSI) and continuing using that pattern. Even with a small number of disks used from a SAN, the order can change from one Linux startup to the next. For example, if one disk in the sequence becomes unavailable, then all the disks that follow it will shift to a different name in the series. The naming order might change in a way that affects the individual nodes differently, which makes the management of the disks complicated and error-prone.

To produce device names that were the same on different Linux systems and also persistent after rebooting required the use of device names in Linux such as `/dev/disk/by-path`, `/dev/disk/by-id`, or `/dev/disk/by-uuid` that are unambiguous. The problem is that those types of names did not fit into spaces provided in the ASM GUI installer for that purpose. It is possible to use these names with the silent install method, which runs a scripts and uses a response file to complete the installation. The problem with the silent install approach when doing the first installation is that there is no interactive error checking, so if any of the input was unacceptable there is no way to remove a failed installation.

This study employed a workaround for this issue, which was to use the file `/etc/zipl.conf` to set the disks in the same order of discovery at startup using the dasd= parameter. With the order controlled this way, it is possible to use the names of the

partitioned files with confidence that the naming is consistent among the nodes and will not change with a reboot.

When the installation was based on Linux as a guest on z/VM®, the order of the disks is controlled by the order of the definition to the guest, for example in the user directory.

The disks managed by ASM are not visible on `/etc/fstab` and are out of sight, so careful disk management must be done in order to avoid errors.

Installing Oracle Real Application Clusters binaries

To install the binaries, it is important to understand the installation sequence, the various tools that can be chosen to perform the installation, and the disk location and size of Oracle binary files.

Installation sequence

When installing Oracle RAC with ASM, whether it is a new installation or a patch, the sequence for the installation is clusterware first on all the nodes, then the database software on all the nodes, and then the ASM software on all the nodes. Each of these software component installations will be designated as an Oracle home, or $HOME. Therefore, an Oracle RAC installation with ASM will have at least three homes when complete. Whenever the administrator is changing homes, it is necessary to change to the environmental variables that work with that home. To make it easier to work from the command-line prompt, aliases were set up in the **oracle** user profile, so that in one step $HOME could be changed to CRS_HOME or ASM_HOME or RDBMS_HOME, and the value for $PATH and the $SID could be changed at the same time.

The standard approach for Oracle installation and patches is for user **oracle** to run an interactive installer to create the installation, and when the setup is satisfactory the installer builds the software. For Oracle RAC, the installer can apply the installation to all the nodes one after the other. The follow-on step is performed by a user with root authority. This step manually runs scripts that were created during the installation on each node, and then the installer can finish the process.

Command line and Oracle GUI, silent installation

Wherever possible in this study, the Oracle command-line interface was used in order to have maximum control over setup parameters and command execution with scripts. The programs that are available to install and manage Oracle include Oracle Enterprise Manager (OEM or EM), and the Database Configuration Assistant, both of which have GUI interface and are commonly used in production environments.

The Oracle Universal Installer (OUI) was used for installing all the components of the cluster. When repeating installations or installing multiple systems, there is the silent installation method that uses scripts created during an installation. The silent installation method does not do error checking along the way, so it is a good idea to do a manual installation first and get immediate feedback if the input is not acceptable, or else risk having to uninstall the first failed attempt.

Oracle installation

The Oracle binaries including clusterware are placed in regular, non-shared Linux files such as EXT3.

Choose a location for the CRS home that is large enough to grow based on the instructions with the package, and taking into account the need for temporary space for patches. The amounts of disk storage recommended by Oracle are conservative, and Oracle home files can grow very large when patches and upgrades are applied.

After constant removal of logs and offloading of files that are not needed, our home files for a single database consisted of this list of files and their sizes:

- `/product/crs 2.5 GB`
- `/product/rdbms 2 GB`
- `/product/asm 2 GB`

This list includes one major patch update and does not have all the automated management tools. A good approach would be to use Linux Logical Volume Manager to create expandable file systems for the binaries and product files. For this installation, a 6.5 GB disk mounted at the root level was used, and the Oracle HOMES were on that disk as directories `/crs`, `/rdbms`, and `/asm`.

Oracle clusterware (CRS)

The first step in the installation of Oracle Real Application Clusters is to install the Oracle clusterware. This is by far the longest and most complex step in the installation process.

Running the Cluster Verification Utility

Oracle clusterware includes a checking program named the Cluster Verification Utility (CVU). The CVU must be run and complete successfully before Oracle Real Application Clusters can be installed. The command to run the CVU is **cluvfy**. The CVU must be run on all of the cluster nodes by both a user with root authority, and by the user **oracle**, before installing clusterware.

The CVU does not check all conditions that would lead to a failure, but the clusterware cannot be installed on a system that is not able to pass the CVU pre-installation checks. These checks include ssh user equivalency, and that the shared storage for the OCR and Voting Disks are correct. CVU should also be run after the installation to verify the success of the installation. The installation steps in the next section show the resources needed to set up clusterware and to pass the CVU pre-installation test.

The following example shows the syntax to use the CVU, where `/product/crs` is the CRS_HOME and **db-node1** and **db-node2** are the servers in the cluster being created:

$CRS_HOME./bin/runcluvfy.sh stage -pre crsinst -n db-node1,db-node2 -verbose

This is a short excerpt from a sample output from running the CVU, where the user **root** failed the check for user equivalence on **db-node1**.

```
Performing pre-checks for cluster file system
Checking node reachability...
Check: Node reachability from node "db-node1"
  Destination Node                      Reachable?
  ------------------------------------  ------------------------
  db-node1                               yes
Result: Node reachability check passed from node "db-node1".
Checking user equivalence...
Check: User equivalence for user "root"
  Node Name                             Comment
  ------------------------------------  ------------------------
  db-node1                              failed
Result: User equivalence check failed for user "root".
ERROR:
User equivalence unavailable on all the nodes.
Verification cannot proceed.
```

When the CVU runs successfully, it is safe to start installing Oracle clusterware. Complete all of the tasks in this list, in the order listed here.

The CVU has capabilities way beyond the scope of this discussion. During the life of a database or cluster, the CVU can be used to evaluate the status and health of the entire installation or one of its components. The related documentation is in the Oracle publication *Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide 10g Release 2 (10.2)*, which is listed in Appendix. References.

Installing Oracle RAC with the Oracle Universal Installer

To start the Oracle Universal Installer (OUI) with the GUI interface on Linux on System z, complete these steps:

1. If running from a Windows system, open a VNC server such as TightVNC, to get a graphical X-Windows interface. More information about VNC is available online on the IBM developerWorks website at:
   http://www.ibm.com/developerworks/linux/library/l-share1.html

   If running from another Linux system, use ssh with the -X parameter to enable X11 forwarding (or enable X11 forwarding in the /etc/ssh/ssh_config file).

2. Log in Linux as user **oracle**.
3. Start OUI with the command ./runInstaller from the directory where the Oracle RAC 10.2.0.2 with clusterware product was unpacked.

   There can be many versions of the runInstaller command on any system, but they are specific to the product and version being installed.

   The Linux access bits of the installer might need to be changed in order to make it executable.

4. For a new installation, the first step is to decide where to put the Oracle inventory file, named oraInventory, so that it is outside of the Oracle HOMEs, and therefore safe from being overwritten during an Oracle RAC upgrade.

   A typical choice is for the location of the oraInventory file is the /opt/oracle/oraInventory directory, not part of any Oracle HOME.

5. Continue the installation by completing the steps in the next section.

Choosing OCR, the Voting Disk, and their backups

Continuing the Oracle RAC installation by defining parameters for the OCR and Voting Disk.

1. The OUI GUI now prompts for information used to build the cluster, by adding one node at a time. Type the IP addresses and interface names and types for each node:
   - `public`
   - `vip`
   - `private`
2. Verify that this information is correct for each node.
3. Type the location of the Oracle Cluster Registry (OCR) and the Voting Disk locations. Both of these are used exclusively by the cluster software to manage the cluster activity.

   These files of 100 MB of disk space can be whole disks, and must be accessible from all the nodes in the cluster. In Oracle RAC 10.2.0.2, both OCR and Voting Disks can be raw devices, block devices, or files on a shared file system such as OCFS2. It is possible to combine any of those three alternatives for OCR and Voting Disk, while having data disks managed by ASM.

4. OUI then prompts for optional shared storage for backups of the OCR and Voting Disks. To create a backup of either requires 100 MB of disk space.

   For this study, the redundancy was ignored and external redundancy was selected.

5. Continue the installation by completing the steps in the next section.

Installing Oracle clusterware 10.2.0.2

Remember that Oracle clusterware 10.2.0.2 must be installed first, and then Oracle clusterware can be upgraded to the 10.2.0.4 version. At this point, the system is ready for Oracle clusterware 10.2.0.2 to be installed. Complete these steps:

1. When doing the initial installation of 10.2.0.2 an error message appears while the clusterware is being installed on each of the servers. This error message refers to client.shared.lib of a makefile. To correct this problem, download patch 6007358 from the Oracle Technical Network (OTN) and open it.
2. Rename the file ~/lib/libclient10.a in order to retain a copy.
3. Copy the module libclient10.a from the patch into the library.
4. Repeat the previous step for the lib32/libclient32.a library.
5. The Linux commands look like this sample, where CRS_HOME is set to /product/crs:

```
cd /product/crs/ lib/
mv lib/libclient10.a   lib/libclient10.a-copy
cp /patchesR1/6007358/files/lib/libclient10.a .
cd /product/crs/lib32/
mv lib32/libclient10.a   lib32/libclient10.a-copy
cp /patchesR1/6007358/files/lib32/libclient10.a
```

6. Click **Retry**, and the automated part of the installation continues until it encounters and resolves the same problem on all of the nodes.
7. When prompted, **leave the OUI session open**, and obtain a session on the first node's public interface.
8. Log in as a user with root authority.
9. Run ~OraInventory/orainstRoot.sh
10. Repeat the last two steps for node2, and any other cluster nodes.
11. Run $CRS_HOME/root.sh

    These scripts must run successfully in order for the clusterware installation to work successfully, although there can be warning messages issued about files not belonging to root.

12. When the root scripts have completed, return to the last panel on the OUI session that was left open in Step 7.
13. Sign off, which will cause the messages and logs from the installation to be displayed.
14. Review these logs and save a copy for future reference and problem determination.
15. There are additional manual steps (see CRS installation manual steps: changing the srvctl and vipca files and CRS installation manual steps: running the vipca program) to be done, however it is acceptable to immediately apply the patch to upgrade Oracle clusterware from 10.2.0.2 to 10.2.0.4. Apply this patch by completing the steps in the next section.

Upgrading to Oracle clusterware 10.2.0.4

Open patch p681089 and start the Oracle Universal Installer (OUI) that is part of the 10.2.0.4 patch package, in a session for the OUI GUI interface. The process is very similar to the installation of Oracle clusterware 10.2.0.2. See Installing Oracle RAC with the Oracle Universal Installer for the exact installation sequence. However, some fields are displayed already having been filled in. Repeat the procedure with the OUI and the root scripts.

When this upgrade is complete, run the manual installation steps by completing the next two sections.

CRS installation manual steps: changing the `srvctl` and `vipca` files

The following manual modification must be made to the binary files `$HOME_CRS/bin/vipca` and `$HOME_CRS/srvctl`:

1. Find this line in each file:
   ```
   export LD_ASSUME_KERNEL
   ```
2. Type in a new line following it:
   ```
   unset LD_ASSUME_KERNEL
   ```
3. In both vipca and srvctl, verify that the lines look like this sample:
   ```
   export   LD_ASSUME_KERNEL
   unset LD_ASSUME_KERNEL
   ```
4. When the RDBMS software is installed, it is also necessary to make the same changes to srvctl in `$HOME_RDBMS/bin/srvctl` in every server node.

For future reference, `srvctl` must be updated after patches are installed.

CRS installation manual steps: running the `vipca` program

It is possible that the `root.sh` script run in Installing Oracle RAC with the Oracle Universal Installer does not use the public interface. This problem can be resolved by running the vipca program manually in a separate session. However, **do not exit from the main OUI session**.

The vipca program starts a new session, which you can use to run the short GUI program vipca. When the task is completed, return to the OUI session, where root can rerun the `root.sh` script and then complete the installation with OUI.

When you have completed these tasks, the next step is to install the Oracle relational database (RDBMS). See Installing the Oracle relational database.

Installing the Oracle relational database

The second step in the installation of Oracle Real Application Clusters is to install the Oracle relational database (RDBMS).

Installing RDBMS and the ASM file system can be done together using the Oracle installer, or the database can be installed on all the nodes, then ASM can be installed on all the nodes, starting with installation of 10.2.0.2 and then upgrading to 10.2.0.4. For this study, the RDBMS installation was done without the installation of ASM, because it was decided to install ASM as a separate step after putting RDBMS on both the nodes.

It is possible to install the base level with OUI, then the upgrade with OUI software, then run the `root.sh` script once on each node, and then do the manual steps only once. To perform the RDBMS installation, complete these steps:

1. When installation package `10202_zlinux_database.zip` is unpacked, open the directory named `Disk2`, which has a version of the runInstaller program.
2. Start runInstaller.
3. On the first panel, select **Enterprise Edition for Linux on System z**.
4. Select **Cluster installation** to install identical software on all of the nodes that can be selected from a list.
5. A panel named **Product-Specific Checks** issues messages on the readiness of the system to accept the installation. If there are any problems, you must resolve them before continuing.
6. At this point in the RDBMS installation, it is possible to create a database and use the installer to define table spaces and data structures. However, for this study a database was created later using scripts, and the installer was not used for database creation.
7. The details of the proposed installation of RDBMS are displayed, and they can be accepted or changed.
8. Similar to the clusterware installation (see Installing Oracle clusterware 10.2.0.2), during the installation of the binaries there is an error message. The problem can be resolved by following the instructions for a manual change to the lib and lib32 libraries.
   Copy the same new versions of `lib/libclient10.a` and `lib32/libclient32.a` to the binary directory, and then click **Retry** as described in Installing Oracle clusterware 10.2.0.2.
9. The installation runs and then displays a window instructing you to run the `root.sh` script on each of the nodes, in the same order that was used during the clusterware installation.
10. The sequence ends with a return to the OUI after running the root scripts.
11. Sign off, which causes the messages and logs from the installation to be displayed.

12. Review these logs and save a copy for future reference and problem determination.

When you have completed these tasks, the final step of the Oracle RAC installation is to install the Oracle Automatic Storage Manager (ASM). See Installing the Oracle Automatic Storage Manager.

Installing the Oracle Automatic Storage Manager

The third and last step in the installation of Oracle Real Application Clusters is to install the Oracle Automatic Storage Manager (ASM).

To install ASM, complete the following steps:

1. Start the runInstaller program in the same manner as the RDBMS installation. However, the first selection is **Configure Automatic Storage Manager (ASM)**. See Installing the Oracle relational database.
2. Select the option to create a new diskgroup, which is a named group of shared disks. For this study, all the shared disks for data storage were put in one diskgroup, named **disks1**. The installer displays every possible partitioned DASD as a candidate for shared storage.
3. When the disks are selected in Linux, ASM is ready to take over their management.

Note:

1. Selecting disks for shared storage for the database can be done after clusterware is installed.
2. Additional disks can be added, at any time. To the user their storage appears as expandable storage, fully striped.

Customize the Oracle Real Application Clusters system

After installing Oracle Real Application Clusters on IBM System z, and installing database binaries and ASM, it is necessary to create listeners and customize the configuration necessary to run a database.

On completion of the Oracle RAC installation, the Oracle Universal Installer (OUI) gives you the option to run the Database Configuration Assistant (DBCA). This tool can be used to set up listeners, set up the dedicated or shared server process preferences, and perform other configurations tasks.

This study bypassed the DBCA and worked directly with the Oracle parameters and settings to perform the customization tasks.

The Oracle pfiles or spfiles are the collection of parameters used when a database is started. pfiles were often used without converting them to spfiles, which are binary but in effect the information applies to either.

The buffer management was handled by Oracle automatically by specifying the **sga_target** parameter.

Setting up the listeners

The listeners for requests for service are created on each node with a `listener.ora` file. A sample is provided in `$HOME_RDBMS/admin/`, but there is no default listener. The sample shown here has two listeners. One listens for requests on the local public vip interface, and the other listens on the other node's public vip interface.

As nodes join the cluster, they register dynamically with the listeners on all of the nodes in the cluster, creating a scenario in which every node can listen for all of the public interfaces on the system. The failover process enhances the availability feature of Oracle RAC, and cross registration provides the framework for server process load balancing. The `listener.ora` file for Oracle Real Application Clusters requires the listener to be configured to recognize both the Oracle interface names and the IP addresses.

This is an example for the listener setup on Oracle RAC node1:

```
(DESCRIPTION_LIST =
   (DESCRIPTION =
      (ADDRESS_LIST =
         (ADDRESS = (PROTOCOL = TCP)(HOST = rac-node1vip)(PORT = 1521))
         )
         (ADDRESS_LIST =
            (ADDRESS = (PROTOCOL = TCP) (HOST = 10.10.10.202) (PORT = 1521))      )
               (ADDRESS_LIST =
                  (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
                  )
            )
          )
       listener_remote_node2 =
          (DESCRIPTION_LIST
             (DESCRIPTION =
                (ADDRESS_LIST =
                  (ADDRESS = (PROTOCOL = TCP)(HOST = rac-node2vip)(PORT = 1521))       )
                  (ADDRESS_LIST =
                     (ADDRESS = (PROTOCOL = TCP) (HOST = 10.10.10.203) (PORT = 1521))
                     )
                     (ADDRESS_LIST =
                        (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
                     )
                  )
              )
```

The combined effect of the connect descriptors, the listeners, and the dedicated or shared servers process (see Client side load balancing) comprises load connection balancing. An essential guide for configuring this is:
Oracle Database Net Services Administrator's Guide 10g Release 2 (10.2) (Number B14212-02)

A link is provided in Appendix. References.

Redo logs and control files

Each instance of the cluster has its own redo logs and control files, which on the test system used for this study, were files managed by ASM. The *redo records* are written to the log files whenever data files are going to be changed, to ensure that in the event of a failure of any kind, the database can be brought back to the last time that it was written to. *Control files* are per instance, and they are essential to open and operate the instance.

Client side load balancing

To build an effective Oracle Real Application Clusters system, balancing the workload coming in from the clients must also be considered.

On the client a `tnsnames.ora` file was set up, and the variable $TNS_ADMIN was set to the location of this file. Sample files came with the Oracle client, located in `client_1/network`.

There are multiple ways to configure for any set up and the elements in the example work with the parameters on the node, and the `listeners.ora` file.

The main element shown here is the connection descriptor for a service named OLTP, which is the database requested - not a particular instance of OLTP but the primary service. The addresses are the public VIPs. The parameter LOAD_BALANCE having been set on causes the client to randomize the use of the addresses that it places the requests on, which is called *server-side load balancing*. In this case, the connect descriptor is handed directly to the service handler on the server that receives it. However it is possible to have a connect descriptor in a `tnsnames.ora` file on the server to resolve what service type is requested.

This example is a `tnsnames.ora` file used in the study:

```
OLTP =
  (DESCRIPTION =
    (ADDRESS_LIST=
    (LOAD_BALANCE=on)
    (FAILOVER=off)
       (ADDRESS = (PROTOCOL = TCP)(HOST = 10.10.10.202)(PORT = 1521))
       (ADDRESS = (PROTOCOL = TCP)(HOST = 10.10.10.203)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = oltp )
    )
  )
```

On the server side, in Oracle Real Application Clusters there are two types of service handlers *servers* and *dispatchers*, which generate server processes for requests handed off by listeners. In dedicated server mode, the server process (and its memory usage) is held sequentially by service requesters until the service is completed and the server process released by the requester. In contrast, in shared server mode dispatchers are the service handlers, serving server processes to a queue of requests which connect and disconnect and reconnect to service requests.

The mechanics of the dedicated or shared server process scenario are determined by the combination of the parameter SERVER=dedicated or SERVER=shared in the connection descriptor that is in effect for the service request, along with the DISPATCHERS pfile or spfile file on the database. If the SERVER parameter is not set, shared server configuration is assumed. However, the client uses a dedicated server if no dispatchers are available.

Workload

The workload used in this study emulated a business processing transactions generated by many active users. These users accessed a single database to process warehouse orders, manage inventory, and bill customers.

Workload description

The workload was configured with a zero amount of think time, the pauses between events that occurs when humans are interactively starting transactions. This results in an intensive load from a single user, so that with the 20 -100 users used in the test environment the workload processed is similar to a much larger number of connected users in a real setting. The level of utilization could be compared to that of a database working as the backend of an application server, or a transaction concentrator.

Workload characteristics

The workload used in this study emulated a small number of warehouses, to intentionally create cluster contention on the warehouse table. This forced a high level of cluster internal communication used to ensure cache consistency and lock handling. This allowed good monitoring of the impact of the various setup modifications.

Performance tools used to analyze the workload

The study used the Oracle AWR reports as a starting point for performance analysis. If contention or excessive latencies are taking place, they show up in the Top 5 wait events as a high percentage of the run time, and they can be analyzed further with the detailed information such as the relevant wait events or the cache information.

SADC and SAR were the primary tools used to evaluate the Linux performance.

The netstat -s command was used for reports of I/O re-sends, to ensure that the TCP/IP communication is not limited by the interface capacity.

Results

This section presents the results of the Oracle RAC study. Two different file systems were compared. Various parameters were altered in an attempt to improve throughput. A test was also done with zEnterprise 196 hardware.

Cluster file system OCFS2 versus Oracle Automatic Storage Management

This test compares the Linux cluster file system OCFS2 with the Oracle Automatic Storage Management (ASM).

The intention of this test is to compare these two cluster storage management tools with regards to the cluster management, and not with regards to the disk I/O bandwidth that they might provide. Therefore, the test runs with a high cache hit ratio to avoid data-related disk I/O contention. These tools use very different ways to provide shared storage as a base for Oracle RAC, and this test shows their impact on transactional throughput. Also, these tests show the characteristics of the workload, including transaction throughput, CPU utilization, and disk and network I/O.

Figure 3 shows the normalized transaction throughput, when scaling users.

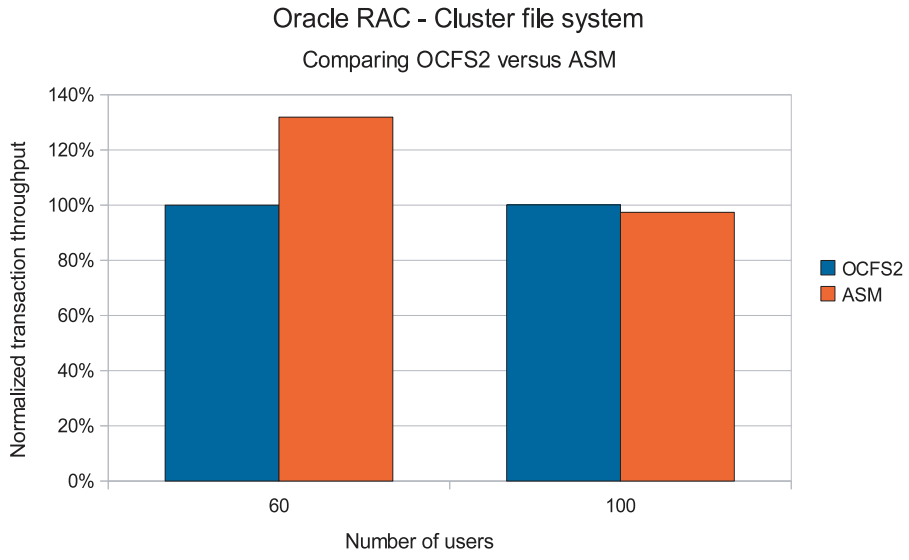

Figure 3. Comparing transaction throughput for cluster file management systems OCFS2 versus ASM

Observation
With 60 users, the transaction throughput with ASM was 32% better than with OCFS2. With 100 users, the comparison reversed and the transaction throughput was 3% better with OCFS2 than with ASM.

Conclusion
The significantly higher transaction rate with ASM with 60 users in the less contended scenario shows an advantage of ASM compared to OCFS2. With 100 users, the contention in the cluster is the dominant factor, which has a greater impact than differences in the underlying tool used to manage the shared storage.

Figure 4 shows the CPU utilization from both nodes represented by the quantity of IFLs used from both nodes.

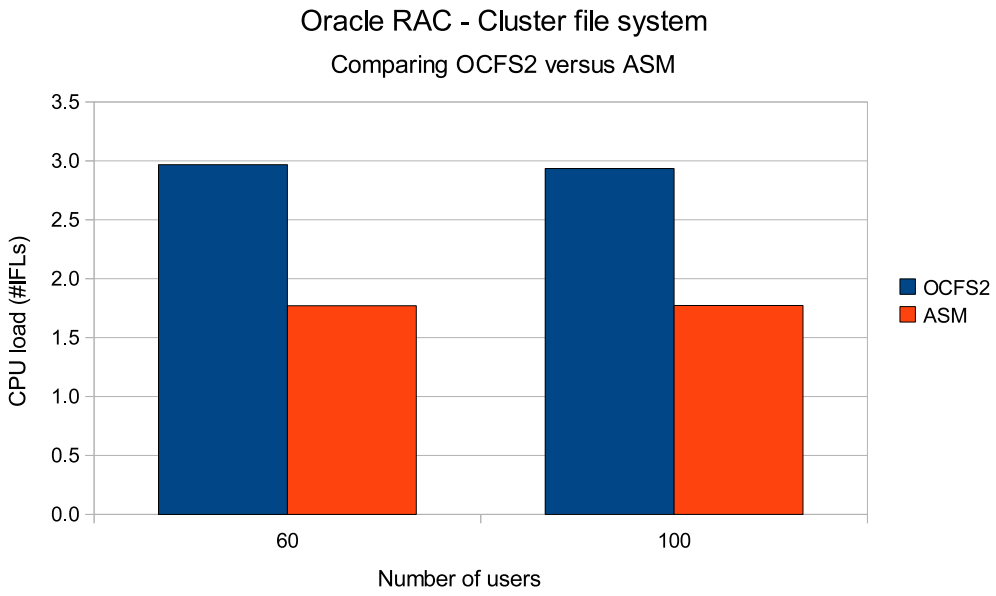

Figure 4. Comparing CPU load for cluster file management systems OCFS2 versus ASM

Observation
When comparing 60 users with 100 users, the CPU load was nearly constant. OCFS2 requires a higher utilization of CPU in absolute terms, approximately three processors compared to a value less than two processors for ASM (four CPUs are available).

Conclusion

The significantly lower CPU utilization with ASM makes the Oracle Automatic Storage Manager a good choice combined with the good transaction throughput reported above.

Disk throughput

Table 5 shows the disk throughput from both nodes.

Table 5. Disk throughput from both nodes using OCFS2 or ASM to manage the shared disk storage

Disk throughput for node 1 and node 2				
Operation	Read (MB/sec)		Write (MB/sec)	
Number of users	60	100	60	100
OCFS2	0.2	0.2	7.4	7.2
ASM	0.3	0.1	8.1	7.4

Observation

The table shows disk storage traffic in megabytes per second in a comparison for OCFS2 and ASM. There is nearly no difference between OCFS2 and ASM, only for the write throughput for the 60 users scenario. The read throughput values are very small, the write values are very moderate.

Conclusion

As expected with a cache hit ratio of 99.9%, the disk I/O values are low. The write throughput follows the transactional throughput and is caused by log I/O and data updates.

Network throughput

Table 6 shows the network throughput from one node.

Table 6. Network throughput for one node using OCFS2 or ASM to manage the shared disk storage

Network throughput - sent and received from one node				
Traffic	Interconnect (MB/sec)		External OSA card (MB/sec)	
Number of users	60	100	60	100
OCFS2	55	55	0.2	0.2
ASM	51	57	0.3	0.2

Observation

The interconnect traffic for the environment with OCFS2 remained the same, while it starts at a lower level and increases when using ASM. The amount of traffic from the communication between the client and Oracle RAC nodes is very low.

Conclusion

It seems that the ASM is correlated with a lower cluster contention than OCFS2 (indicated by the lower interconnect traffic), which increases when the number of users working in parallel increases.

The synchronization of the shared cache uses database blocks, meaning that even the smallest update statement causes a full block exchange over the interconnect when the data is accessed from both nodes. That phenomenon is what the data ratio between client communication and interconnect shows very impressively. In this scenario a default database blocksize of 8 KB was used.

Cluster Contention - default block size

This test shows the impact on transaction throughput when reducing the default block size from 8 KB to 4 KB.

The results from Cluster file system OCFS2 versus Oracle Automatic Storage Management showed that a large default database block size might be a drawback when using Oracle RAC with shared tables, because the global cache is managed in units of the database block size.

Large blocks cause higher traffic for the interconnect, and might increase the chances of a lock collision.

Figure 5 shows the impact on transactional throughput and wait classes when decreasing the default block size from 8 KB to 4 KB.

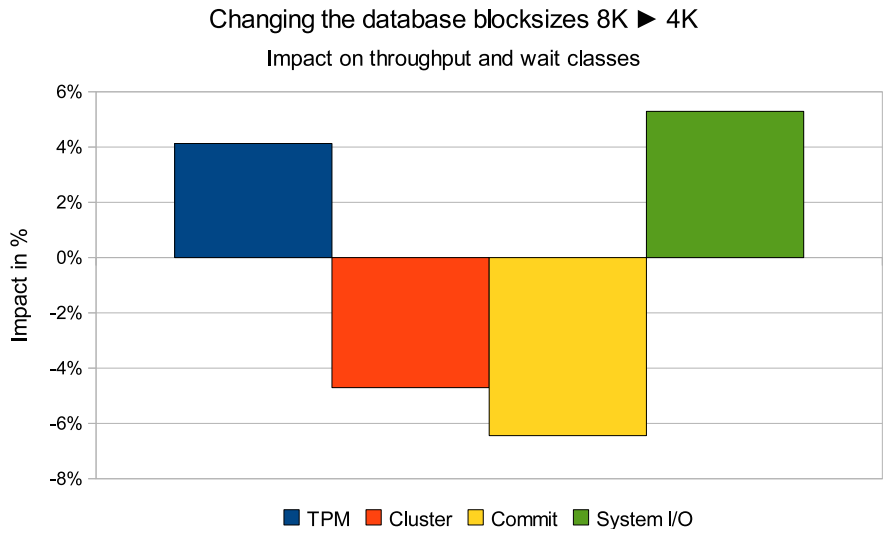

Figure 5. Impact of decreasing the database block size on transaction throughput and time spent in wait events

Observation

When the Oracle database default block size was decreased from 8 KB to 4 KB, the transaction throughput improved by 4%. The data shown describes classes of waits in which cluster commit and System I/O are the most significant portions. There was 4.71% less waiting due to cluster contention compared to the 8 KB block size. The commit contention was reduced by 6%. The system I/O as a percent of total activity went up 5%, corresponding to the increased throughput, which causes more log file I/O and data updates.

Figure 6 shows the impact on the global cache profile when decreasing the database block size from 8 KB to 4 KB.

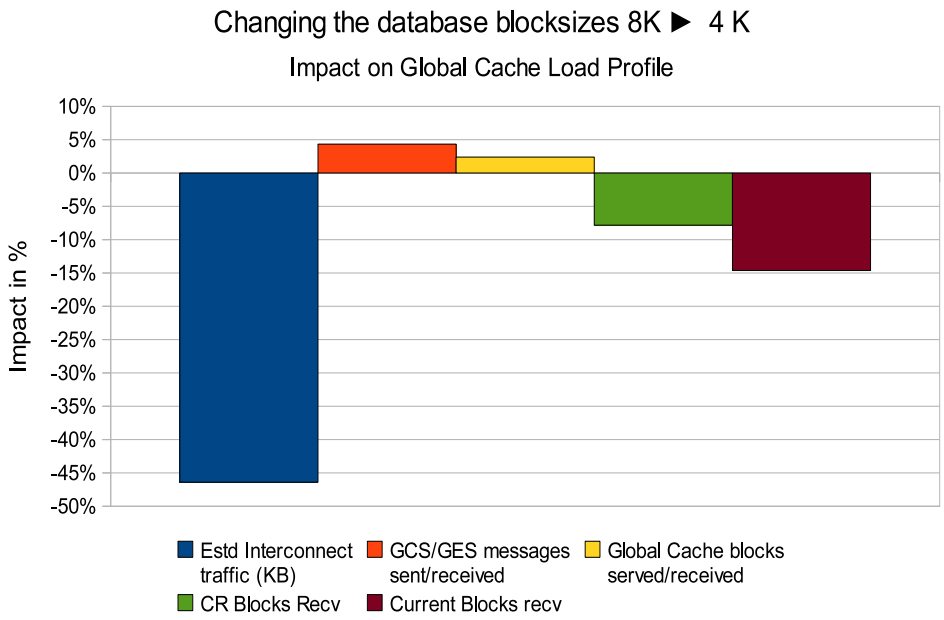

Figure 6. Impact of decreasing the database block size on the global cache profile

Observation

When the database default block size was reduced from 8 KB to 4 KB, the amount of traffic over the interconnect to perform cache fusion went down by 46%. The Global Cache Load Profile from the AWR reports from both instances gives a breakdown of the types of traffic, which is composed of both data blocks and messages. Shown here is the increase in both types of data. GCS/GES messages increased 4% in units of messages sent and received, and Global Cache blocks sent and received increased 2%. GES stands for Global Enqueue Services and GCS stands for Global Cache Services, both are important for performance of the cache system.

CR Blocks Received went down 8%, and Current Blocks Received went down by 15% The distinction between CR blocks and Current Blocks is whether they are consistent reads across the cluster or Current or "as is" for the latest.

Conclusion

The reduction of the database block size leads, as expected, to a decrease of the interconnect traffic of approximately one-half. The transaction throughput follows the cluster wait events (less contention means more throughput), while the commit contention is reduced even more, which indicates a reduced lock contention. The quantity of messages sent and received increase as required by the higher throughput, but the number of data block is reduced. This

reduction clearly shows that the contention on the blocks has been reduced, because locks for the two different 4 KB blocks in the same 8 KB block are now two independent locks, making the sending of the block unnecessary (but not the communication that a lock must be placed).

The performance improvement as measured by transaction throughput leads to the recommendation of using the smallest possible block size (such as 4 KB), as default in an Oracle RAC system with shared tables.

Dedicated server versus shared server processes

This test shows the impact on transaction throughput when changing between Oracle dedicated server processes and Oracle shared server processes.

With dedicated server processes, the CPU load increased to the full system utilization. Therefore, another test was done with an additional CPU for each Oracle RAC node. The Oracle RAC dedicated server and shared server processes are described in the section.

Figure 7 shows the transaction throughput when switching from shared to dedicated Oracle server processes.

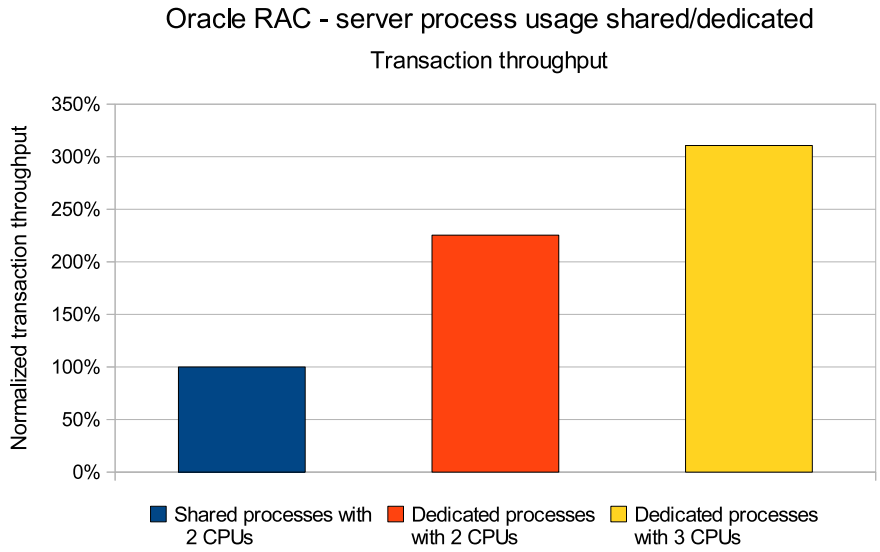

Figure 7. CPU load for switching the usage of the Oracle server processes from shared to dedicated

Observation
By changing the Oracle RAC server processes to dedicated from shared, transaction throughput increased by 125%. Adding a third processor to each CPU increased transaction throughput another 86% compared to the measurement of the shared server processes with 2 CPUs.

Conclusion
This test scenario demonstrates that higher throughput can be obtained with dedicated server processes configured for the Oracle RAC servers compared with shared server processes. With this change, the CPU load increased to the full system utilization. Adding CPU capacity in both Oracle RAC servers improves the throughput with dedicated server processes further.

This effect is contributed to our workload with a moderate number of clients (up to 80), where each client is 100% active, generating a continuous workload, which easily keeps one Oracle server process busy. If several of these workload generating clients are sharing a server process, this server process becomes a bottleneck.

For this type of workload pattern, the use of dedicated server processes is recommended. This workload pattern is typically produced by a batch job, an application server, or a connection concentrator.

The opposite scenario occurs with connections that have a low utilization, such as from any manual interactions, and with a very high number (1000 or more) of users. Here, the use of dedicated server processes results in thousands of under-utilized server processes, with the corresponding memory requirements.

Therefore, it is important to see how memory usage scales, when the number of users with dedicated server processes increases. Figure 8 shows the sum of the amount of memory used from both nodes, as reported from the **kbmemused** value from the sadc and sar data.

Note: If the SERVER parameter is not set, then shared server configuration is assumed. However, the client uses a dedicated server if no dispatchers are available.

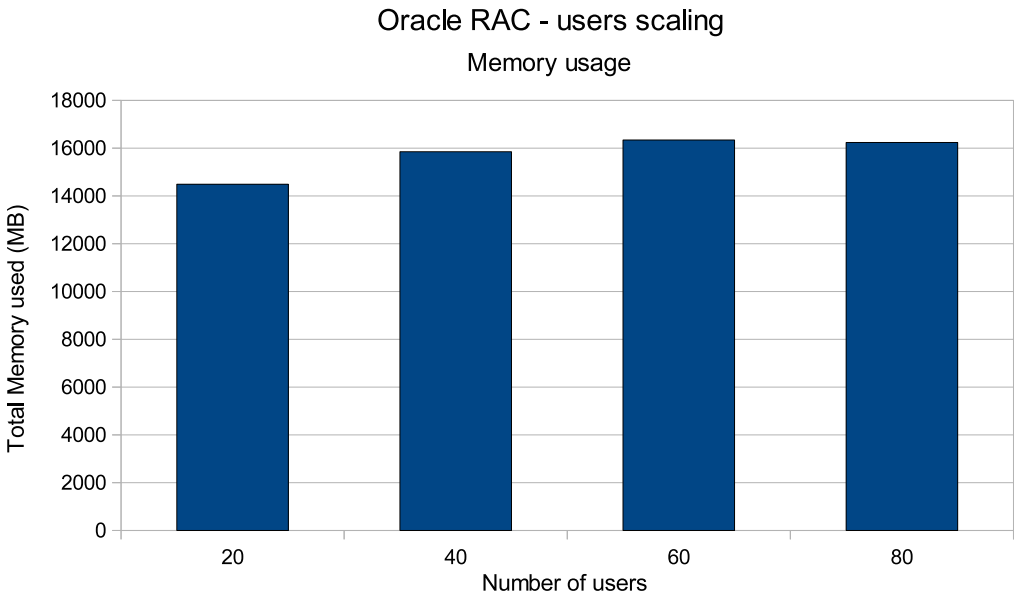

Figure 8. Memory usage when scaling workload generating users (100% active) with dedicated Oracle server processes

Observation

With the first increment going to 40 users from 20, the memory usage increased from approximately 14 GB to 15.6 GB (approximately 11%). The next increment of 20 users caused a slight increase of overall memory usage, and with the next increment the used memory size stays constant. Each Oracle RAC node has 16 GB memory, which makes a total of 32 GB from which about 16 GB are in use, indicating that plenty of free memory is available in both systems.

Conclusion

This chart shows that after overall memory usage increases to accommodate 40 users, the total usage levels off. One parameter that limits the increase of the memory utilization is the fixed size of the SGA, which seems to be fully utilized with 60 users. Another factor that might explain a decreasing memory requirement is increasing contention for locks. The amount of memory needed for additional server processes is not necessarily limited by these parameters because it is in the first place Linux operating system memory, not managed by Oracle.

It seems that the operating system memory needed for these processes soon becomes constant because of optimizations from the operating system. An example of this Linux optimization is the memory used for the binaries. This memory is used only one time, regardless of the

number of copies. Only data structures, unique to each process, occupy individual pages. The memory related to the user data is kept in the SGA (which has a limited size). All these factors explain why the amount of memory used does not increase indefinitely.

Figure 9 shows how the two topmost wait events are developing when the number of users is scaled.

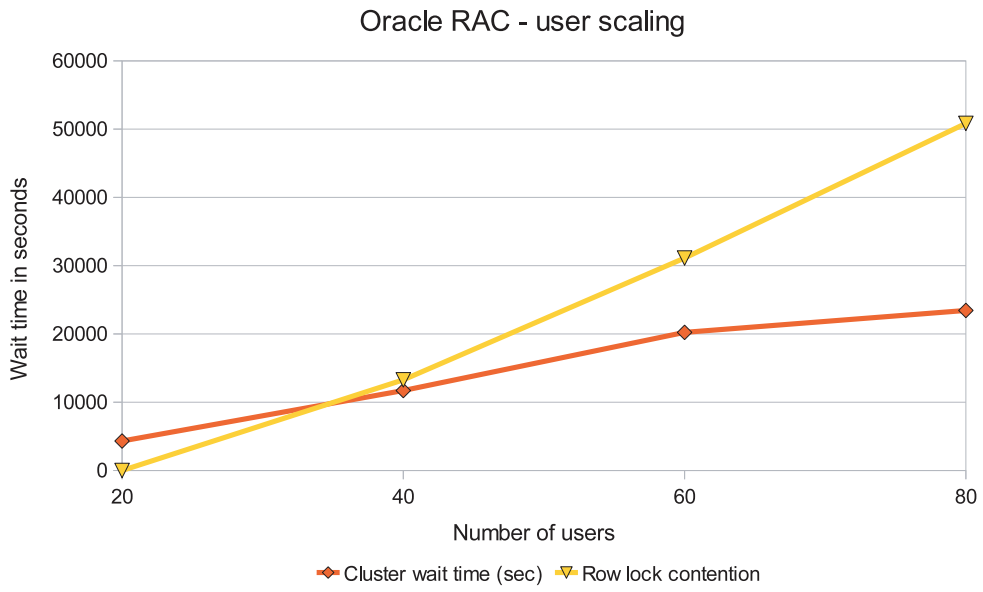

Figure 9. Cluster wait time and lock contention when scaling workload generating users (100% active) with dedicated Oracle server processes

Observation

This graph shows two types of contention taking place on the clustered system as users were scaled from 20 to 80. The cluster wait time (indicating lock contention between the nodes) trends up from with 20 users and the slope decelerates and begins to flatten with 60 users. The lock contention, which is related to lock contention inside one node, starts at nearly 0 and increases much faster than the cluster contention. The break-even point is at 40 users.

Conclusion

The behavior of the system is determined by two types of lock contentions, either globally between the two nodes or locally inside each node. The local lock contention

becomes the dominant factor when the number of users is increasing. The lock contention is a bottleneck that must be resolved if user activity is to be increased. In this case, the cause of lock contention is updates made to a small shared table. Lock contention occurs either when the nodes in the cluster, or two users inside the nodes, are trying to update the same data. The local contention is based on row locking, while the cluster contention locks the whole data block.

The graph shows also that the typical measurement point at 40 users demonstrates a certain level of cluster contention (which makes it worthwhile to consider network tuning of the interconnect). At the same level there is a contention on local locks, which keeps the workload realistic and means that there is not a 100% dependency from the cluster performance.

Network Setup - buffer counts

This test shows the impact on network throughput when increasing the inbound buffer count value.

The inbound buffer count is a Linux network interface parameter that can be increased to allow more packets received in flight, in order to increase throughput. This parameter is defined per network device individually. The default value is 16.

Figure 10 shows the impact on transaction throughput when increasing the inbound buffer count parameter from the default value of 16 to 64 and 128.

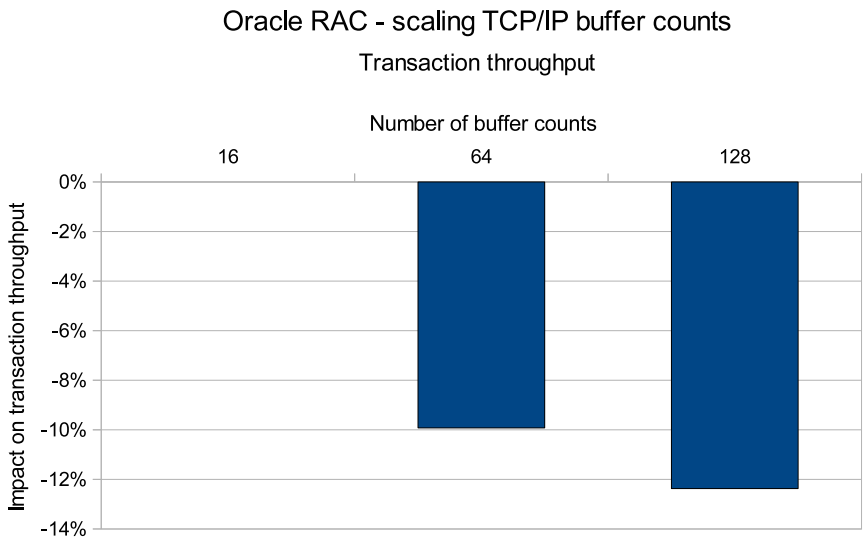

Figure 10. Impact on transaction throughput when scaling the inbound network buffer count size

Observation

When increasing the amount of inbound buffers from the default value of 16 to 64 and then to 128, the throughput degrades for each step. An inbound buffer count of 64 buffers resulted in a 10% reduction of transaction throughput. An inbound buffer count of 128 resulted in an additional 12% reduction of transaction throughput.

Conclusion

The best choice for the test environment of this study is the default inbound buffer count of 16.

Note: Increasing the inbound buffer count is typically recommended for optimum network throughput. In this scenario, the higher network throughput led to a reduced transaction rate, because it is correlated with an increased cluster contention and increased communication using the Interconnect needed to maintain the cache consistency.

Network Setup - MTU size

This test shows the impact on transaction throughput when changing the Maximum Transmission Unit size.

The Maximum Transmission Unit (MTU) size is the maximum packet size in bytes that a network interface can send over a network device. This size includes the protocol

information, such as the TCP and IP headers. This size does not contain the Ethernet header. This study is concerned with the MTU settings that can be set in the configuration files in SLES, for example, in `/etc/sysconfig/network/ifcfg-qeth-bus-ccw-0.0.0xxx` for each device. The effective MTU size is also negotiated during the connection creation with the other end point, and the MTU can be reduced to the capacity of that device.

When the MTU is smaller than the packet size, the data must be divided into MTU sized packets. On the receiving side the packets must be reassembled, thus creating overhead that scales with the quantity of packages required. When the MTU is only slightly too small, there can also be wasted bandwidth. For example, when the MTU is slightly smaller than the average packet size the secondary packet can be almost empty.

An MTU size bigger than the packet size should have no impact on the network bandwidth, if all intermediate nodes on the path to the final destination support the MTU size, because only the real data are sent. But the number of packages processed in parallel is calculated with the TPC/IP windows size and the MTU size. When the window size stays constant and the MTU size is increased, this number is decreased, which might lead to a performance degradation, when the smaller MTU size would be appropriate as well.

Therefore, the ideal is to find the MTU size that fits to the package size used by the application and is supported by all intermediate nodes on the path to the final destination.

This study used as MTU sizes 1492 or 8192. For Jumbo frames, an MTU size of 8992 would be also possible, but 8192 is considered to be more memory efficiently because it fits exactly in two 4 KB pages, while 8992 would need three. With HiperSockets on IBM System z, the MTU size is implicitly defined by the MFS (maximum frame size) set on the IOCDS parameter. The MTU size for HiperSockets with an MFS size of 16 KB is 8192 bytes.

With Oracle RAC, the public interfaces are used to communicate with the clients for transactions and the private network is used as the interconnect between the nodes for messages and cache blocks.

The interface for the Oracle interconnect exchanges cached database pages between nodes, therefore we expect large packets, where a large MTU size might improve performance of the entire cluster.

Figure 11 depicts the impact on transaction throughput when scaling the MTU sizes of the LAN connection and the Interconnect compared to an MTU size of 8192 for both.

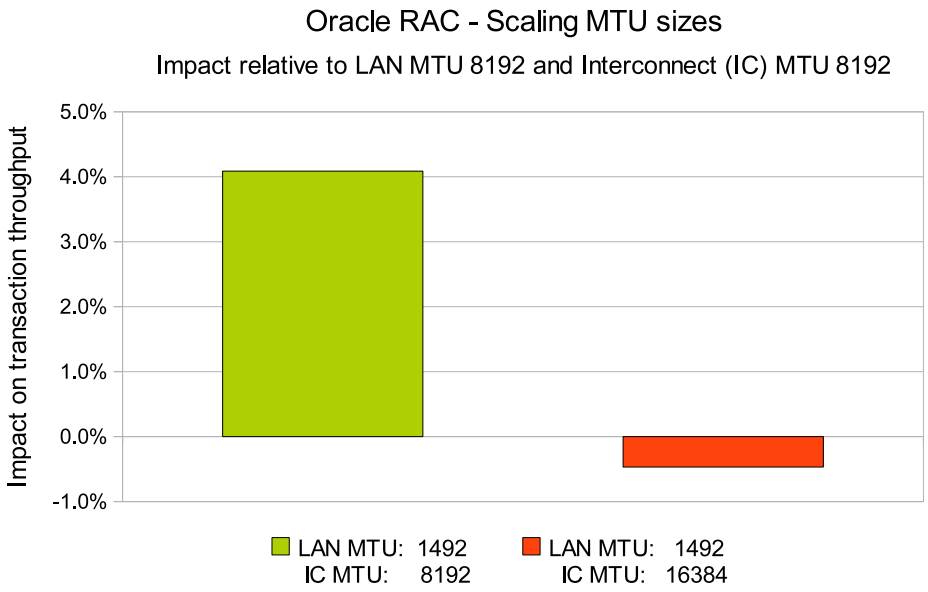

Figure 11. Impact on transaction throughput when scaling the MTU sizes of the LAN connection and the Interconnect compared to an MTU size of 8192 for both (LAN: external network connections to the clients, IC: interconnect between the RAC nodes)

Observation

The baseline example that these tests are measured against are with public LAN Ethernet interface set at 8192 and the private interface which was HiperSockets set at 8192. The first variation shows the impact of modifying the 1 Gb OSA interconnection to 1492, resulting in an improvement of 4.1%. The second example shows the private interface being set with MTU scaled up to 16384, with the result being a decrease in transactional throughput. The average packages size for send or received was 1290 -1330 bytes in all scenarios.

Conclusion

It seems that either Oracle RAC or the Oracle client use as maximum package size 1330 bytes for all TCP/IP connections. Increasing the MTU size without increasing the TCP/IP window size leads to the effect that fewer packages are held in the TCPIP stack, which explains why the best throughput is seen with the small MTU sizes (1492 for the LAN and 8192 for the Interconnect).

Network setup - device queue length

This test shows the impact on network throughput when increasing the device queue length.

The Linux parameter net.core.netdev_max_backlog is one of the TCP/IP tuning parameters that controls the number of received packets that can be queued on Linux for a network interface. This parameter is a system-wide parameter, and influences the receive behavior for all interfaces. This parameter can be modified dynamically with the syntax:

```
sysctl -w net.core.netdev_max_backlog=2000
```

This parameter can also be modified permanently in the `/etc/sysctl.conf` file.

The default value for this parameter on Novell SLES 10 is 1000. The experience is that a larger value improves the performance of many applications. Figure 12 shows the impact on the transaction throughput when increasing this parameter from the default of 1000 to 2000 and then 2500.

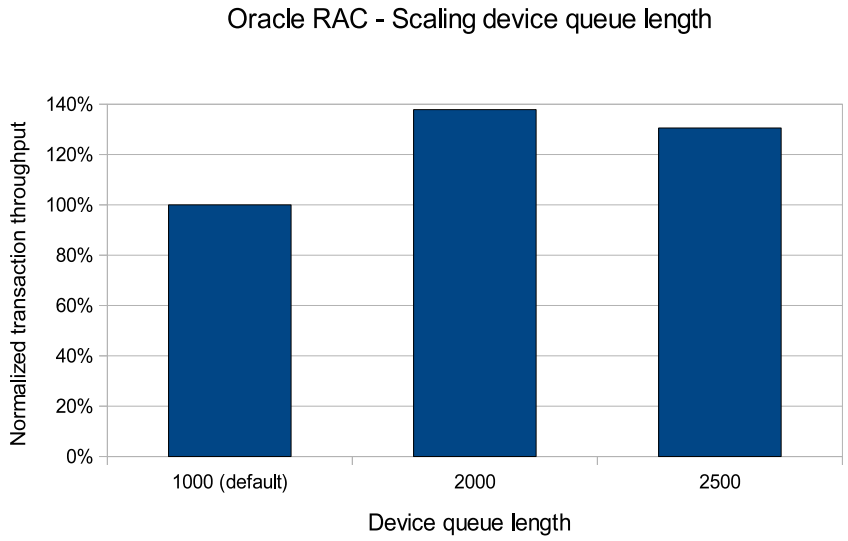

Figure 12. Normalized transaction throughput when scaling the device queue length of the network devices

Observation

Increasing the device queue length from 1000 to 2000 leads to a throughput improvement of

approximately 40%. Increasing the device queue length further, to 2500, results in a moderate throughput degradation.

Conclusion

The device queue length is an important tuning parameter. An improvement of approximately 40% is a very high value. But the observation that there is a size that is too large, resulting in degradation, indicates the need for careful usage of this parameter. Increasing this parameter is best done in combination with monitoring of the network or transaction throughput.

Network Setup - interconnect network device type

This test shows the impact on network traffic throughput when using a HiperSockets connection for network traffic, instead of a 1 Gb or 10 Gb OSA card.

As shown in Cluster file system OCFS2 versus Oracle Automatic Storage Management, the network traffic over the interconnect is very much higher than the communication traffic with the clients. Therefore, the bandwidth of the underlying network device type is an important performance parameter.

Figure 13 shows the improvement of using HiperSockets compared to a 10 Gb OSA card or a 1 Gb OSA card.

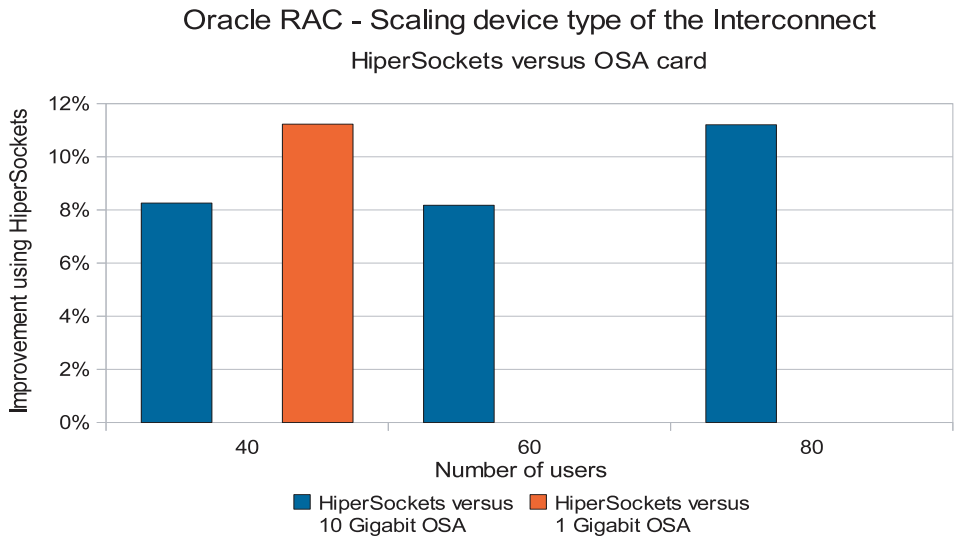

Figure 13. Improvement when using HiperSockets or the Interconnect versus 1 Gb or 10 Gb OSA card

Observation

This graph shows the relative improvement in transactional throughput when HiperSockets is compared with two other types of devices used for the Cluster Interconnect. With 40 users, the improvement is 8% when compared to a 10 Gb OSA card and 11% when compared to a 1 Gb OSA card. At higher workload levels, the advantage of the HiperSockets connection increases to 11% when compared to a 10 Gb OSA card.

Conclusion

HyperSockets are the best choice for the Oracle RAC interconnect. It is possible to use HiperSockets when the server nodes are on the same physical IBM System z. The second choice would be to use a 10 Gb OSA card. As shown here, the use of a 10 Gb OSA connection does not necessarily require a full 10 Gb infrastructure with switches, and so forth. A direct coupling of two OSA ports with a standard cable makes the connection in

point-to-point mode, which additionally reserves the full bandwidth for the interconnect and avoids any interaction with other network workloads, ensuring a constant response time.

In combination with the 1 Gb OSA connection a new wait event occurs, wait event **cr request retry**, indicating that the 1 Gb interconnect is not sufficient for the workload processed.

IBM zEnterprise 196

This test shows the improvement on our workload when switching from an IBM System z10 to an IBM zEnterprise 196.

Figure 14 shows the impact on the transactional throughput driven per CPU utilization.

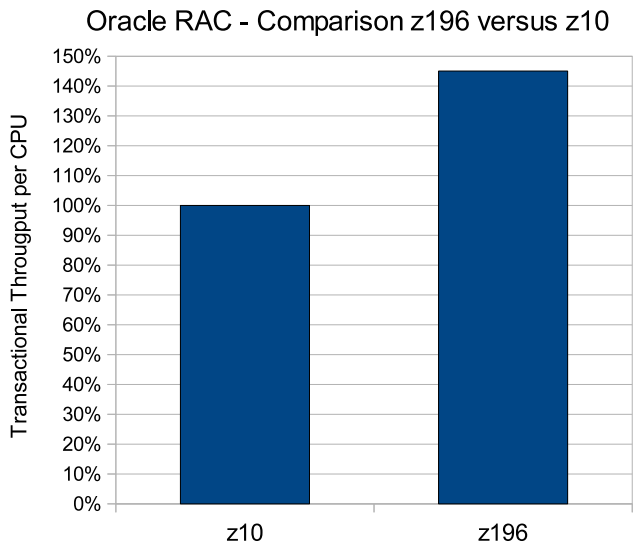

Figure 14. Improvement of transactional throughput per CPU utilization when upgrading from IBM System z10 to IBM zEnterprise 196

Observation

The upgrade to IBM zEnterprise 196 provides a significant improvement of the throughput and a reduction of the CPU load. This change improves the throughput driven per CPU by 45%.

Conclusion

The new IBM zEnterprise 196 provides a impressive improvement of the workload driven per CPU, which makes it is very attractive for that workload. These results are comparable with other Linux on IBM zEnterprise 196 results, for example the results shown at: http://www.ibm.com/developerworks/linux/linux390/perf/tuning_z10.html#z196

Be aware that the intended contention inside the cluster (see Cluster Contention - default block size) is one parameter that limits additional improvements due solely to faster processor speed.

Appendix. References

These references provide more information about Oracle Real Application Clusters, the IBM hardware used in this study, specific topics addressed in this study, and topics of related interest.

- Oracle publishes extensive documentation for its products at this Oracle website:
  http://www.oracle.com/pls/db102/gateways
- Clusterware on Linux
  Oracle Clusterware and Oracle Real Application Clusters Installation Guide 10g Release 2 (10.2) for Linux (Number B14203-09).
  Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide 10g Release 2 (10.2) (Number B14197-15)
- Oracle Net Services
  Oracle Database Net Services Administrator's Guide 10g Release 2 (10.2) (Number B14212-02)
  http://download.oracle.com/docs/cd/B19306_01/network.102/b14212/toc.htm
- Oracle RAC 10g Release 2
  http://www.oracle.com/pls/db102/gateways
- Oracle Database 10g Release 2 (version 10.2.0.2) for Linux on IBM System z
  http://www.oracle.com/technetwork/database/enterprise-edition/downloads/index.html
- Oracle on Linux for IBM System z
  IBM Redbooks publication: Experiences with Oracle Solutions on Linux for IBM System z (IBM SG24-7634-00)
  http://www.redbooks.ibm.com/abstracts/sg247634.html
- IBM disk storage
  IBM Redbooks publication: DS8000 Performance Monitoring and Tuning (IBM SG247146-01)
- Oracle Technical Network (OTN)
  http://www.oracle.com/technetwork/database/enterprise-edition/downloads/index.html
- Setting up OCFS2 is described in this white paper:
  http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/index.jsp?topic=/liaag/l0wocf00_2010.htm

Copyright IBM Corporation 2011
IBM Systems and Technology Group
Route 100
Somers, New York 10589
U.S.A.
Produced in the United States of America,
03/2011
All Rights Reserved

IBM, IBM eServer, IBM logo, developerWorks, DS8000, ECKD, FICON, HiperSockets, Redbooks, S/390, System Storage, System p, System x, System z System z10, zEnterprise, z/OS and z/VM are trademarks or registered trademarks of the International Business Machines Corporation.

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

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

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

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.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

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

ZSW03185-USEN-02