

# **DB2 10 for Linux on System z Using z/VM v6.2, Single System Image Clusters and Live Guest Relocation**

**Maximize DB2 10 availability using SSI  
clusters and LGR**

**Discover the benefits of  
z/VM 6.2 LGR**

**Use DB2 STMM and  
z/VM 6.2 LGR**



**Lydia Parziale  
Dan Davenport  
Eduardo Kienetz  
Tito Ogando  
Manoj Srinivasan Pattabhiraman  
Srivatsan Venkatesan**





International Technical Support Organization

**DB2 10 for Linux on System z Using z/VM v6.2, Single  
System Image Clusters and Live Guest Relocation**

November 2012

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

**First Edition (November 2012)**

This edition applies to Version 6, Release 2 of z/VM and DB2 Version 10, Release 1 for Linux on System z

**© Copyright International Business Machines Corporation 2012. All rights reserved.**

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

# Contents

<b>Notices</b> .....	v
Trademarks .....	vi
<b>Preface</b> .....	vii
The team who wrote this book .....	vii
Now you can become a published author, too! .....	viii
Comments welcome .....	ix
Stay connected to IBM Redbooks .....	ix
<b>Chapter 1. Overview of z/VM 6.2 and DB2 v10 autonomic features</b> .....	1
1.1 Virtualization overview .....	2
1.1.1 z/VM introduction .....	2
1.2 z/VM 6.2 features .....	2
1.2.1 Single system image .....	2
1.2.2 Live guest relocation .....	3
1.3 Autonomic features - DB2 version 10 .....	4
1.3.1 Self-Tuning Memory Management .....	4
1.3.2 Automatic storage databases .....	4
1.3.3 Configuration Advisor .....	5
1.3.4 Automatic statistics collection .....	5
1.3.5 Automatic reorganization .....	5
1.3.6 Utility throttling .....	5
<b>Chapter 2. Our lab environment</b> .....	7
2.1 Overview of our four-member cluster .....	8
2.2 Overview of our two-member cluster .....	9
2.3 Applications used for testing .....	10
<b>Chapter 3. General memory management</b> .....	11
3.1 Linux swap .....	12
3.2 General memory configuration parameters .....	13
3.2.1 User limit requirements .....	13
3.2.2 Kernel parameters .....	14
3.3 Large page support .....	16
3.3.1 Database shared memory size configuration parameter .....	17
3.3.2 Enabling large page support .....	17
3.3.3 Testing relocation while DB2 is using large pages .....	18
3.4 z/VM memory sizing considerations .....	19
3.4.1 Memory and paging considerations for live guest relocation .....	19
<b>Chapter 4. DB2 Self-tuning memory manager and z/VM live guest relocation</b> .....	21
4.1 DB2 Self-tuning memory manager .....	22
4.2 Determining DB2 memory configuration .....	22
4.2.1 Fetching instance memory configuration .....	22
4.2.2 Fetching database memory configuration .....	23
4.2.3 Enabling the self-tuning memory manager .....	24
4.2.4 Verification of enablement .....	24
4.3 DB2 LGR on a memory-constrained target cluster member .....	25
4.3.1 Verifying database sort heap size and bufferpool page size .....	27

4.3.2 Varying database sort heap and buffer pool size . . . . .	28
4.3.3 Verifying new database configuration . . . . .	29
4.3.4 Load generation queries . . . . .	29
4.3.5 Analysis of the DB2 memory tracker report . . . . .	30
4.3.6 Relocating a Linux guest to a memory-constrained cluster member . . . . .	32
4.3.7 Verification of z/VM health after the relocation . . . . .	32
4.3.8 Load generation and database memory setup . . . . .	33
4.4 Test conclusions . . . . .	37
<b>Chapter 5. Time synchronization . . . . .</b>	<b>39</b>
5.1 Hardware clock . . . . .	40
5.1.1 Server time protocol (STP) . . . . .	40
5.1.2 Relocating DB2 when using the hardware clock . . . . .	41
5.2 Network Time Protocol (NTP) . . . . .	52
5.2.1 Configuring NTP . . . . .	52
5.2.2 NTP and different hardware clocks . . . . .	54
5.3 Final Results . . . . .	54
<b>Chapter 6. Networking . . . . .</b>	<b>57</b>
6.1 System diagram of a two-system cluster . . . . .	58
6.2 Examining how virtual local area networks (VLANs) behave . . . . .	58
6.3 VSWITCH MAC ID . . . . .	59
6.4 Changing to jumbo frames and relocating DB2 . . . . .	64
6.4.1 Environment . . . . .	65
6.4.2 Tests . . . . .	65
<b>Chapter 7. Best practices for DB2 v10 during live guest relocation . . . . .</b>	<b>69</b>
7.1 Processor sharing . . . . .	70
7.2 Dynamic processor allocation - share settings . . . . .	70
7.2.1 Share setting scenario . . . . .	70
7.3 Memory considerations . . . . .	71
7.3.1 z/VM paging . . . . .	72
7.3.2 Linux swapping . . . . .	72
7.4 Networking consideration . . . . .	72
7.4.1 Intraensemble data network . . . . .	73
<b>Related publications . . . . .</b>	<b>75</b>
IBM Redbooks . . . . .	75
Other publications . . . . .	75
Online resources . . . . .	75
Help from IBM . . . . .	76
<b>Index . . . . .</b>	<b>77</b>

# Notices

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

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.*

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

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

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

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

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

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

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


## COPYRIGHT LICENSE:

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

# Trademarks

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

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

DB2®	System Storage®	z/VM®
HiperSockets™	System z®	z/VSE®
IBM®	WebSphere®	z10™
Redbooks®	z/Architecture®	zEnterprise®
Redbooks (logo)  ®	z/OS®	zSeries®

The following terms are trademarks of other companies:

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

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

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



# Preface

IBM® z/VM® 6.2 introduced significant changes to z/VM with a multisystem clustering technology that allows up to four z/VM instances in a single system image (SSI) cluster. This technology is important because it offers you an attractive alternative to vertical growth by adding new z/VM systems. In the past, this capability required duplicate efforts to install, maintain, and manage each system. With SSI, these duplicate efforts are reduced or eliminated.

Support for live guest relocation (LGR) lets you move Linux virtual servers without disrupting your business or incurring loss of service, thus reducing planned outages. The z/VM systems are aware of each other and take advantage of their combined resources. LGR enables you to relocate guests from a system requiring maintenance to a system that will remain active during maintenance.

A major advantage for DB2® v10 clients is that using z/VM 6.2 does not require any changes to existing DB2 structures. This remarkable benefit is due to the fact that DB2 v10 is installed as part of the Linux guest on z/VM and is fully integrated into LGR. This allows you to smoothly move DB2 v10 when you move Linux virtual servers, without interrupting either DB2 v10 or z/VM operations and services.

This IBM Redbooks® publication will help you understand how DB2 10 on Linux for System z® behaves while running on a z/VM that is being relocated using z/VM's 6.2 Live Guest Relocation feature.

In this book we explore memory management, the DB2 self-tuning memory manager feature, time synchronization, networking, and storage and performance considerations with regard to relocation. We also offer some best practices found during a live guest relocation for DB2 v10.

## The team who wrote this book

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

**Lydia Parziale** is a Project Leader for the ITSO team in Poughkeepsie, New York, with domestic and international experience in technology management including software development, project leadership, and strategic planning. Her areas of expertise include e-business development and database management technologies. Lydia is a certified PMP and an IBM Certified IT Specialist with an MBA in Technology Management. She has been employed by IBM for over 25 years in various technology areas.

**Dan Davenport** is an IT Management Consultant for IBM Lab Services. He is located in Irving, Texas. He has 30 years of experience on System z. His expertise includes Linux on System z, z/VM, performance, and system security. Dan has contributed to many projects upgrading, migrating or converting to z/VM and Linux on System z.

**Eduardo Kienetz** is a Staff Software Engineer at the IBM Linux Technology Center (LTC) in Brazil. He has over 10 years of experience in the information technology field, 2.5 years of which were in the banking area (Canada), having contributed numerous hours to open source projects and presented at some Linux events. He has worked at IBM for a year, and currently leads the LTC Infrastructure' development team. His areas of expertise include Linux systems

administration, integration, and software development. He holds a Computer Science degree from UNIFRA-RS, Brazil.

**Tito Ogando** is an IT Specialist in Brazil. He has four years of experience in Linux on IBM System z. He is currently working as a Linux Specialist for the Linux team in Brazil supporting more than 1800 Linux on System z servers for the IBM internal account. He is also an avid Python programmer responsible for maintaining the main tool for automating server builds and managing workloads of his team. He holds a degree in Computer Science from Universidade Federal da Bahia. His areas of expertise include Linux on IBM System z, troubleshooting, and automation.

**Manoj Srinivasan Pattabhiraman** is a System z Technical Consultant with IBM Singapore. He has more than 12 years of experience in z/VM and Systems Programming for Linux on zSeries®. Manoj is an IBM Certified IT Specialist (Level 2) with a Masters in Computer Application. In his current role, he provides consultation and implementation services for various Linux on System z clients for Growth Markets. Also, he supports the System z Benchmarking Center in Singapore by running benchmarks and PoCs. Manoj has contributed to several z/VM and Linux on System z related IBM Redbooks publications, and has been a presenter at various technical conferences and ITSO workshops for z/VM and Linux on System z.

**Srivatsan Venkatesan** is an IT Specialist in Systems and Technology Group in USA. He has gained one year of experience in z/VM and Linux on System z. He holds a degree in Computer Science from the University of South Carolina. His areas of expertise include Linux and middleware on System z.

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

David Bennin, Roy P. Costa, Alfred Schwab  
International Technical Support Organization, Poughkeepsie Center

Martin Kammerer  
IBM Germany

Antonio Jose Goncalves de Freitas  
IBM Brazil

Thanks to the authors of the supporting IBM Redbooks publications:

- ▶ Authors of *An Introduction to z/VM Single System Image (SSI) and Live Guest Relocation (LGR)*, SG24-8006, published in April 2012, were:  
Lydia Parziale, Anthony Bongiorno, Howard Charter, Jo Johnston, Volker Maen, Clovis Pereira, Sreehar Somasundaran and Srivatsan Venkatesan
- ▶ Authors of *Using z/VM v6.2 single systemimage (SSI) and live guest relocation (LGR)*, SG24-8039, published in September 2012, were:  
Lydia Parziale, Edi Lopes Alves, Klaus Egeler, Susy Heisser, Rob Hunt, Jo Johnston, Volker Masen and Srivatsan Venkatesan

## Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your

network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

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

[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

## Comments welcome

Your comments are important to us!

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

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

[ibm.com/redbooks](http://ibm.com/redbooks)

- Send your comments in an email to:

[redbooks@us.ibm.com](mailto:redbooks@us.ibm.com)

- Mail your comments to:

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

## Stay connected to IBM Redbooks

- Find us on Facebook:

<http://www.facebook.com/IBMRedbooks>

- Follow us on Twitter:

<http://twitter.com/ibmredbooks>

- Look for us on LinkedIn:

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

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

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

- Stay current on recent Redbooks publications with RSS Feeds:

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





# Overview of z/VM 6.2 and DB2 v10 autonomic features

Autonomic computing provides the ability to self-manage and adapt to unpredictable changes in increasingly more complex computer systems. An autonomic system can make decisions on its own through the use of high-level policies, constant checking to optimize its status, and adaptation to changing conditions. IBM DB2 version 10 autonomic features include a number of self-configuring, self-optimizing and self-healing capabilities.

In this chapter, we start by giving a brief overview of virtualization and z/VM. We then discuss the latest features of z/VM v6.2. Finally, in this chapter, we discuss some autonomic features of DB2 v10 and how these features can be combined with z/VM 6.2 virtualization and clustering. By combining both DB2 v10 and the live guest relocation feature of z/VM v6.2, we can eliminate the downtime required for planned hardware maintenance without affecting the integrity of business-critical data.

## 1.1 Virtualization overview

Virtualization is used to make a single physical resource appear to be multiple logical resources. For example, a processor core can appear to function as multiple virtual processors, or multiple storage devices can be consolidated into a single logical pool in order to increase the utilization of the available storage space. Therefore, virtualization makes server deployment and utilization more flexible. You can reduce the costs of administration, power, and floor space by consolidating data servers through virtualization. As an added benefit, you can use virtualization to significantly increase server utilization and improve overall performance efficiency.

### 1.1.1 z/VM introduction

z/VM is an operating system for the IBM System z platform that provides a highly flexible test and production environment. The z/VM implementation of IBM virtualization technology provides the capability to run full-function operating systems such as Linux on System z, z/OS®, z/VSE® and z/TPF as “guests” of z/VM. z/VM supports 64-bit IBM z/Architecture® guests and 31-bit IBM Enterprise Systems Architecture/390 guests.

z/VM provides each user with an individual working environment known as a virtual machine. The virtual machine simulates the existence of a dedicated real machine, including processor functions, memory, networking, and input/output (I/O) resources. Operating systems and application programs can run in virtual machines as guests. For example, you can run multiple Linux and z/OS images on the same z/VM system that is also supporting various applications and users. As a result, development, testing, and production environments can share a single physical computer. A virtual machine uses real hardware resources, but even with dedicated devices (like a tape drive), the virtual address of the tape drive may or may not be the same as the real address of the tape drive. Therefore, a virtual machine only knows virtual hardware that may or may not exist in the real world.

## 1.2 z/VM 6.2 features

z/VM single system image enables live migration of running virtual guests from one LPAR or physical system to another with zero downtime, continuous service availability, and complete transaction integrity. This capability makes hardware maintenance possible at any time of the day and does not require redundant servers. z/VM SSI cluster makes it possible to move online workloads as required from one SSI cluster to another in order to maintain service level agreements and various performance goals.

### 1.2.1 Single system image

The z/VM Single System Image feature (VMSSI) is an optional priced feature that is new with z/VM version 6.2.

A z/VM single system image (SSI) cluster is a multisystem environment in which the z/VM systems can be managed as a single resource pool and guests can be moved from one system to another while they are running. Each SSI member is a z/VM logical partition (LPAR) connected via channel-to-channel (CTC) connections.

A z/VM SSI cluster consists of up to four z/VM systems in an Inter-System Facility for Communications (ISFC) collection. Each z/VM system is a member of the SSI cluster. The cluster is self-managed by the z/VM control program (CP) using ISFC messages that flow

across channel-to-channel devices between the members. All members can access shared DASD volumes, the same Ethernet LAN segments, and the same storage area networks (SANs). Figure 1-1 shows a four-member SSI cluster.

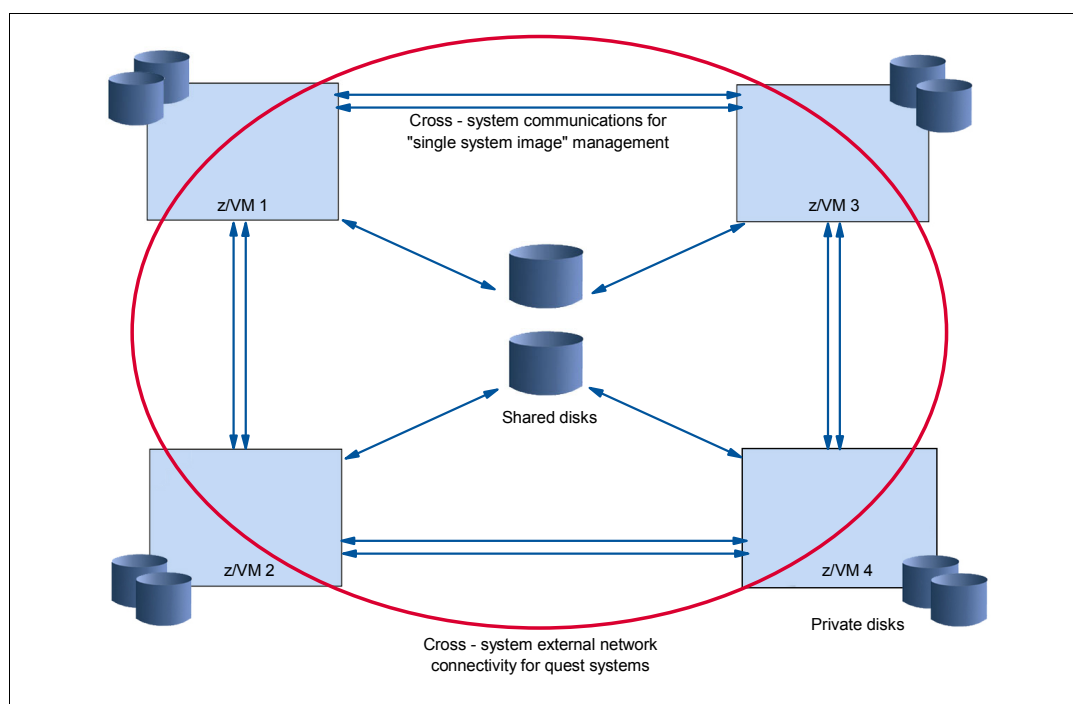


Figure 1-1 An SSI cluster with four members

The IBM z/VM Single System Image feature (VMSSI) enhances z/VM systems management, communications, disk management, device mapping, virtual machine definition management, installation, and service functions to enable up to four z/VM systems to share and coordinate resources within an SSI cluster.

For further details, refer to *z/VM CP Planning and Administration*, SC24-6178, and also to the IBM Redbooks publication, *An Introduction to z/VM Single System Image (SSI) and Live Guest Relocation (LGR)*, SG24-8006, chapters 2, 4 and 5.

## 1.2.2 Live guest relocation

With the IBM z/VM single system image (SSI), a running Linux on System z virtual machine can be relocated from one member system to any other, a process known as live guest relocation (LGR). LGR occurs without disruption to the business. It provides application continuity across planned z/VM and hardware outages, and flexible workload balancing that allows work to be moved to available system resources.

**Operating systems supported:** Linux on System z is currently the only guest environment supported for relocation.

There are several reasons why you might need to relocate a running virtual server:

- ▶ Maintenance of hardware and/or software
- ▶ Fixing performance problems
- ▶ Workload rebalancing

Relocating virtual servers can be useful for load balancing and for moving workload off a physical server or member system that requires maintenance. After maintenance is applied to a member, guests can be relocated back to that member, thereby allowing you to maintain z/VM as well as keeping your Linux on System z virtual servers available.

## 1.3 Autonomic features - DB2 version 10

The DB2 Version 10 database server is the fastest-growing, flagship database offering by IBM. The DB2 product works seamlessly in the System z virtualization environment, straight out of the box. DB2 recognizes and reacts to any dynamic relocation events, such as runtime changes to the computing and physical memory resources of a host partition.

It comes equipped with host dynamic resource awareness, automatic features such as self-tuning memory management (STMM), and enhanced automatic storage, which greatly reduces administrative overhead for tuning and maintenance. These functions make the DB2 product well suited for the virtualization environment and enable it to leverage the System z virtualization technology.

### 1.3.1 Self-Tuning Memory Management

The Self-Tuning Memory Management feature automatically adjusts and redistributes DB2 heap memory in response to dynamic changes in partition memory and workload conditions. This memory-tuning feature simplifies the task of memory configuration by automatically setting values for several memory configuration parameters. When enabled, the memory tuner dynamically distributes available memory resources among the following memory consumers: buffer pools, package cache, locking memory, and sort memory.

The tuner works within the memory limits defined by the `database_memory` configuration parameter. The value of the parameter itself can be automatically tuned as well. When self-tuning is enabled for `database_memory` (when you set it to `AUTOMATIC`), the tuner determines the overall memory requirements for the database and increases or decreases the amount of memory allocated for database shared memory depending on the current database requirements. For example, if the current database requirements are high and there is sufficient free memory on the system, more memory will be consumed by database shared memory. If the database memory requirements decrease or if the amount of free memory on the system becomes too low, some database shared memory is released.

### 1.3.2 Automatic storage databases

An automatic storage database is one in which table spaces can be created and whose container and space management characteristics are completely determined by the DB2 database manager. At the most basic level, databases that are enabled for automatic storage have a set of one or more storage paths associated with them. A table space can be defined as “managed by automatic storage” and its containers assigned and allocated by DB2 based on those storage paths.

A database can only be enabled for automatic storage when it is first created. You cannot enable automatic storage for a database that was not originally defined to use it; similarly, you cannot disable automatic storage for a database that was originally designed to use it.

DB2 creates an automatic storage database by default. The command line processor (CLP) provides a way to disable automatic storage during database creation by explicitly using the `AUTOMATIC STORAGE NO` clause.



### 1.3.3 Configuration Advisor

When you create a database, this tool is automatically run to determine and set the database configuration parameters and the size of the default buffer pool (IBMDEFAULTBP). The values are selected based on system resources and the intended use of the system. This initial automatic tuning means that your database performs better than an equivalent database that you could create with the default values. It also means that you will spend less time tuning your system after creating the database. You can run the Configuration Advisor at any time (even after your databases are populated) to have the tool recommend and optionally apply a set of configuration parameters to optimize performance based on the current system characteristics.

### 1.3.4 Automatic statistics collection

Automatic statistics collection helps improve database performance by ensuring that you have up-to-date table statistics. The database manager determines which statistics are required by your workload and which statistics must be updated. Statistics can be collected either asynchronously (in the background) or synchronously, by gathering runtime statistics when SQL statements are compiled. The DB2 optimizer can then choose an access plan based on accurate statistics. You can disable automatic statistics collection after creating a database by setting the database configuration parameter `auto_runstats` to OFF. Real-time statistics gathering can be enabled only when automatic statistics collection is enabled. Real-time statistics gathering is controlled by the `auto_stmt_stats` configuration parameter.

### 1.3.5 Automatic reorganization

After many changes to table data, the table and its indexes can become fragmented. Logically sequential data might reside on nonsequential pages, forcing the database manager to perform additional read operations to access data. The automatic reorganization process periodically evaluates tables and indexes that have had their statistics updated to see if reorganization is required, and schedules such operations whenever they are necessary.

### 1.3.6 Utility throttling

This feature regulates the performance impact of maintenance utilities so that they can run concurrently during production periods. Although the 22 Database Administration Concepts and Configuration Reference impact policy for throttled utilities is defined by default, you must set the impact priority if you want to run a throttled utility. The throttling system ensures that the throttled utilities run as frequently as possible without violating the impact policy. Currently, you can throttle statistics collection, backup operations, rebalancing operations, and asynchronous index cleanup. For more information, refer to *DB2 V10.1 Database Administration Concepts and Configuration Reference*.





## Our lab environment

In our lab environment, we set up two z/VM SSI clusters. One was a four-member SSI cluster, the other a two-member SSI cluster. In this chapter, we provide a short overview of both clusters. For a more detailed description of both clusters, see the IBM Redbooks publication *An Introduction to z/VM Single System Image (SSI) and Live Guest Relocation (LGR)*, SG24-8006.

## 2.1 Overview of our four-member cluster

The four-member SSI cluster is named ITSOSIA and consists of members ITSOSI1, ITSOSI2, ITSOSI3, and ITSOSI4, as shown in Figure 2-1.

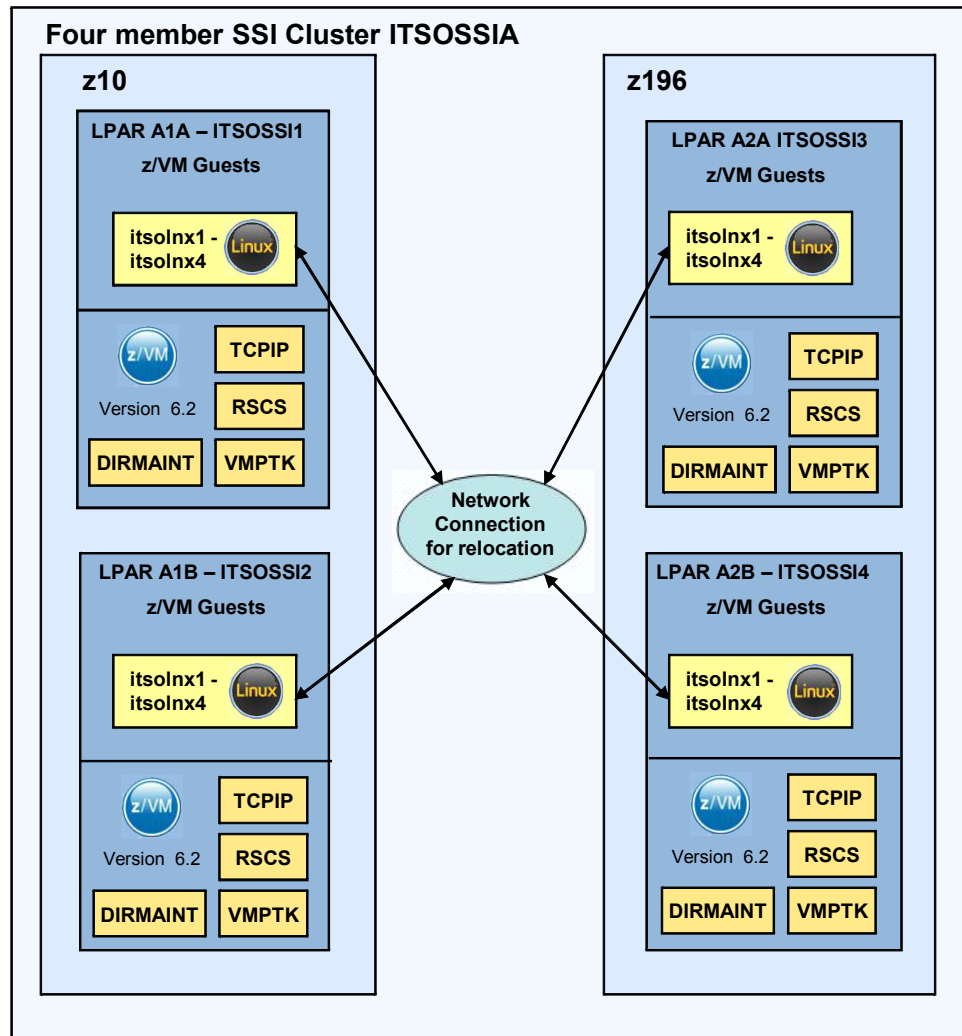


Figure 2-1 Our four-member cluster.

### TCP/IP configuration

The TCP/IP configuration of the members is as shown in Table 2-1.

Table 2-1 z/VM TCPIP setup for our four member cluster

Hostname	ITSOSI1	ITSOSI2	ITSOSI3	ITSOSI4
Domain Name	itso.ibm.com	itso.ibm.com	itso.ibm.com	itso.ibm.com
IP Address	9.12.4.232	9.12.4.233	9.12.4.234	9.12.4.235
Memory	30 GB	8 GB	30 GB	8 GB

## Linux on System z guests

Table 2-2 lists information about the Linux on System z guests that are candidates for relocation.

Table 2-2 Linux on System z guest information on our four-member cluster

Hostname	ITSOLNX1	ITSOLNX2	ITSOLNX3	ITSOLNX4	ITSOLNX5
Domain Name	itso.ibm.com	itso.ibm.com	itso.ibm.com	itso.ibm.com	itso.ibm.com
IP Address	9.12.4.141	9.12.4.140	9.12.4.227	9.12.4.228	9.12.5.116
Memory	4 GB	4 GB	6 GB	6 GB	6 GB
OS	SLES11 SP 1	SLES 11	SLES 11	RHEL 5.6	SLES 11

## 2.2 Overview of our two-member cluster

We used two clusters with two members in each.

The first two-member cluster is named POKLBS. The members are named POKLBS1 and POKLBS2, as shown in Figure 2-2.

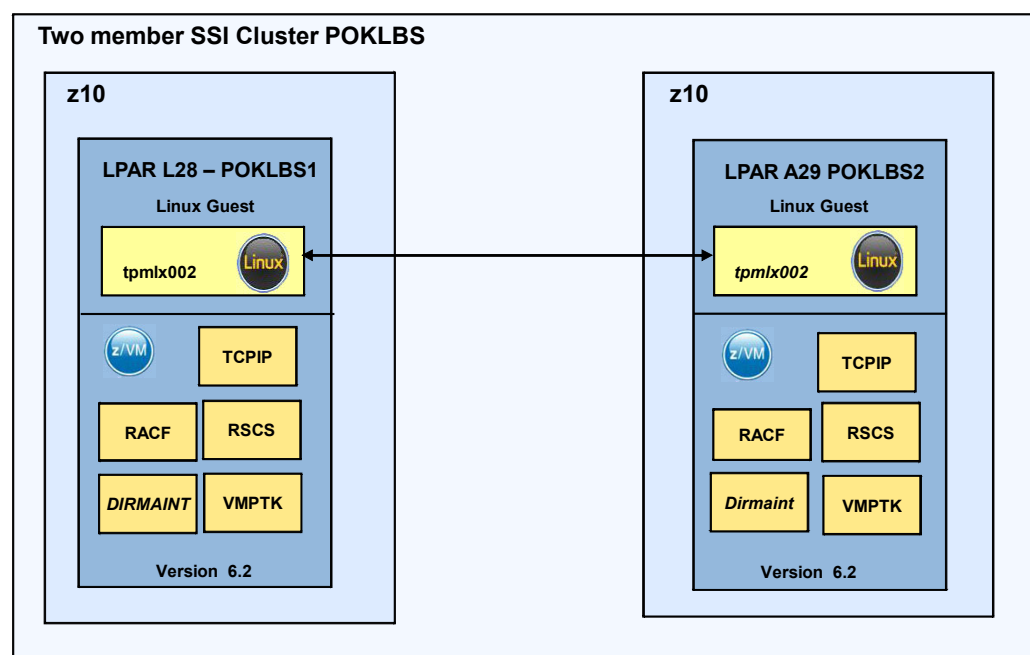


Figure 2-2 Two-member cluster

### TCPIP configuration

The TCPIP configuration of our two-member cluster is shown in Table 2-3.

Table 2-3 z/VM TCPIP setup for our two members cluster

Hostname	POKLBS1	POKLBS2
IP Address	9.12.22.250	9.12.22.251
Memory	14 GB	14 GB

## Linux guests

Table 2-4 shows the Linux on System z guests that are eligible for relocation on the cluster.

*Table 2-4 Linux guest information on our two-member cluster*

Hostname	tpmlx002
IP Address	9.12.22.51
Memory	4G
OS	SLES11 SP2
Additional applications	DB2

## 2.3 Applications used for testing

The applications we installed on the Linux on System z guests in our SSI clusters to produce a load on the system when we were relocating the Linux guests using live guest relocation are as follows:

- ▶ IBMDB2 UDB V10.1

We installed DB2 v10 following the standard installation instructions for Linux on System z.

- ▶ IBM WebSphere® Application Server V7.0.
- ▶ The IBM Trade Performance Benchmark sample

A stocks and shares benchmarking application that runs on WebSphere Application Server and uses DB2 v10.1 to store its data. This benchmark provides a real world workload, driving WebSphere's implementation of J2EE 1.4 and Web services, including key WebSphere performance components and features. Trade 6 simulates a stock trading application that allows you to buy and sell stock, check your portfolio, register as a new user, and so on. We used Trade 6 to generate workloads that we were then able to analyze in terms of their impact on system performance during a live guest relocation.

- ▶ We used a workload generator to automatically run transactions on Trade 6 and to put a load on the Linux guest when it was relocated using live guest relocation.



## General memory management

In this chapter we provide information on z/VM memory management parameters that influence DB2 v10 performance. We present minimal and/or required memory management parameter values and whether they are influenced by Linux on System z or z/VM v6.2.

Additionally, we discuss z/VM v6.2 sizing considerations as well as the memory and paging considerations when running a live guest relocation of a z/VM SSI member that has DB2 v10.1 on Linux for System z as a guest.

## 3.1 Linux swap

While the phrases *swap disk* and *swapping* are often used when discussing Linux, Linux does not swap process spaces in the traditional sense but instead does paging based on least recently used algorithms. In this sense, swap is a special format that allows the partition to be used as virtual memory. Swap space is used when an application requires more memory than is physically available. This is a costly operation, so ensure that swap and physical memory together are large enough to host all your processes otherwise Linux will start to kill processes when it is short on memory. To size your Linux on a System z guest, you may want to use a monitor to watch for swapping and shrink the guest size until it starts swapping.

The general swap recommendation for Linux on System z is to keep swapping to a minimum where possible. To keep swapping as fast as possible, you may want to use virtual disks for the first small swapping device. Of course, this is only possible if you have enough main memory available and you have plenty of real storage.

When defining a virtual disk as swap in the z/VM v6.2 environment, you could unintentionally prevent a live guest relocation. If the virtual disk is only used for swap and is not shared, no restrictions exist to use a virtual disk that is defined by the DEFINE command. Changing from MDISK V-DISK to COMMAND DEFINE VFB-512 allows for the relocation. Example 3-1 shows the updated directory statement using the correct commands for live guest relocation.

**Important:** Format DASD and have them defined as SWAP every time the Linux guest is started so that the DASD is emulated in memory more effectively.

*Example 3-1 Directory statement*

---

```
type itsolnx1 direct
USER ITSOLNX1 ITSOSI 4G 4G G
INCLUDE LINDFLT
COMMAND DEFINE VFB-512 AS 0111 BLK 128000
IPL 300
MACH ESA 4
NICDEF 0600 TYPE QDIO LAN SYSTEM ITSOSVSW1
* MDISK 0111 FB-512 V-DISK 128000
MDISK 0300 3390 0001 10016 LXDE1C MR
MDISK 0400 3390 0001 30050 LX603D MR
Ready; T=0.01/0.01 11:53:50
vmrelocate test itsolnx1 to itsossi3
User ITSOLNX1 is eligible for relocation to ITSOSI3
```

---

See *An Introduction to z/VM Single System Image (SSI) and Live Guest Relocation (LGR)*, SG24-8006 for more information.

A good practice is to prevent the system from swapping by keeping all of your processes in physical memory. Size your DB2 and other applications in such a way that Linux does not use swap.

See *Linux on IBM System z: Performance Measurement and Tuning*, SG24-6926 for more information and performance test results for various swap device options.



## 3.2 General memory configuration parameters

There are at least three types of configuration parameters that directly affect DB2 usage: user process resource limits (ulimits), virtual memory, and interprocess communication. Each of these configuration parameters is discussed in this section.

### 3.2.1 User limit requirements

The `ulimit` command can set (and report) user process resource limits.<sup>1</sup>

Depending on your installation, the DB2 database engine automatically raises the ulimits to varying degrees:

- For root installations, the DB2 database engine automatically raises ulimits where necessary based on the needs of the DB2 database system.
- For non-root installations, the DB2 database engine can update only the data, nofiles, and fsizelimits for the engine process up to the hard limits imposed by the system administrator.

In either case, it might be more practical to set the resource limits permanently on your system. Especially for non-root installations, the data, nofiles, and fsizelimit values should be set appropriately by an administrator after installation.

After a non-root installation is completed, verify the operating system hard ulimits for the data, nofiles, and fsizeresources as the instance owner. The recommended values are listed in Table 3-1.

Table 3-1 Recommended ulimit values for non-root installations

Hard ulimit resource	Description	Minimum value	Recommended value	Command to query the value
<b>data</b>	Maximum private memory allowed for a process	The amount of memory available on the computer	Unlimited	<code>ulimit -Hd</code>
<b>nfiles</b>	Maximum number of open files allowed for a process	Larger than the sum of all MAXFILOP database configuration parameters for all databases in the instance	65536	<code>ulimit -Hn</code>
<b>fsizel</b>	Maximum file size allowed	Unlimited	Unlimited	<code>ulimit -Hf</code>

Parameter definitions:

`ulimit -H` Change and report the maximum limit associated with this resource. A hard limit can only be raised by root so it is useful for security. This prevents a process from overstepping its resources.

`ulimit -n` Sets the maximum number of open files.

<sup>1</sup> See the following website for up-to-date information, under "Operating system user limit requirements (Linux and UNIX)":

<http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/>

ulimit -d        Sets the maximum size of the process's data segment.  
 ulimit -f        Sets the file size limit in blocks.

If the minimum ulimit values are not met, the DB2 database engine could encounter unexpected operating system resource shortage errors. These errors can lead to a DB2 database system outage.

## 3.2.2 Kernel parameters

Set up kernel parameters before you start the DB2 database. The default values may cause issues such as memory starvation when running a DB2 database system <sup>2</sup>. To avoid such issues, *semaphores* and *shared memory* values should be updated. If you are able to install DB2 as root, these parameters may be configured automatically.

### Interprocess communication kernel parameters

The Linux interprocess communication (IPC) kernel parameters allow for multiple processes to communicate with one another.

To improve DB2's performance and memory usage, adjust several of these kernel parameters. By doing so, the database manager prevents unnecessary resource errors. Table 3-2 shows the recommended kernel parameters that should be modified.

Table 3-2 Recommended kernel parameters for modification

Parameter	Description	Recommended Value
kernel.shmmax	Defines the maximum size of one shared memory segment in bytes.	90% of total memory, but if you have a large amount of storage you can leave 512 MB to 1 GB for the operating system instead.
kernel.shmall	Define the available memory for shared memory in 4KB pages.	You should convert the shmmax value to 4 KB. (shmmax value * 1024 / 4)
kernel.shmmni	Define the maximum number of shared memory segments.	4096. This amount enables large segments to be created avoiding the need for thousands of small shared memory segments. This parameter may vary depending on your application.
kernel.sem	Four values must be set in this parameter. The first one is the number of semaphores, the second indicates the maximum number of semaphores. The third is the maximum number of semaphores operations within one semop call. And the fourth limits the number of allocatable semaphores.	250 256000 32 1024
kernel.msgmni	Maximum number of queues on the system	1024
kernel.msgmax	Maximum size of a message in bytes	65536
kernel.msgmnb	Default size of a queue in bytes	65536

<sup>2</sup> See the website for up-to-date information, under "Kernel parameter requirements (Linux)":  
<http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/>

## Modifying the kernel parameters

You must have root privileges to modify kernel parameters.

To update kernel parameters on Red Hat Enterprise Linux and SUSE Linux <sup>3</sup>, do the following:

1. Run the **ipcs -l** command to list the current kernel parameter settings.
2. Analyze the command output to determine whether you must change kernel settings by comparing the current values with the enforced minimum settings shown in Example 3-2. This example shows the output of the **ipcs** command with comments that we added after the **//** to indicate the parameter names.

*Example 3-2 ipcs -l command output*

---

```
ipcs -l

----- Shared Memory Limits -----
max number of segments = 4096           // SHMMNI
max seg size (kbytes) = 32768           // SHMMAX
max total shared memory (kbytes) = 8388608 // SHMALL
min seg size (bytes) = 1

----- Semaphore Limits -----
max number of arrays = 1024             // SEMMNI
max semaphores per array = 250          // SEMMSL
max semaphores system wide = 256000    // SEMMNS
max ops per semop call = 32             // SEMOPM
semaphore max value = 32767

----- Messages: Limits -----
max queues system wide = 1024           // MSGMNI
max size of message (bytes) = 65536     // MSGMAX
default max size of queue (bytes) = 65536 // MSGMNB
```

---

3. Modify the necessary kernel parameters by editing the **/etc/sysctl.conf** file. If this file does not exist, create it. Example 3-3 shows an example of what should be placed into the file.

**Note:** The file **/etc/sysctl.conf** is provided via a package (RPM). On SUSE Linux Enterprise Server 11, you would find it in the PROCPS RPM file, and in Red Hat Enterprise Linux, the INITSCRIPTS RPM file.

*Example 3-3 Kernel parameters within /etc/sysctl.conf*

---

```
#Example for a computer with 16GB of RAM:
kernel.shmmni=4096
kernel.shmmax=17179869184
kernel.shmall=8388608
#kernel.sem=<SEMMSL> <SEMMNS> <SEMOPM> <SEMMNI>
kernel.sem=250 256000 32 4096
kernel.msgmni=16384
kernel.msgmax=65536
```

<sup>3</sup> See the website for up-to-date information, under “Modifying kernel parameters (Linux)”: <http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/>

kernel.msgmnb=65536

---

4. Run the command **sysctl -p** to load the sysctl settings from the default file `/etc/sysctl.conf`:

```
sysctl -p
```

**Optional:** To make the changes persist after every reboot:

- ▶ SUSE Linux - Make `boot.sysctl` active (execute the command **chkconfig sysctl on**, as root).
- ▶ Red Hat Enterprise Linux - The `rc.sysinit` initialization script reads the `/etc/sysctl.conf` file automatically.

For the latest information about supported Linux distributions, see:

<http://www.ibm.com/software/data/db2/linux/validate/>

### Additional kernel parameter settings

Additional kernel parameter settings are listed in Table 3-3.

Table 3-3 Configuring other Linux kernel parameters

Kernel parameter setting	Configuring the kernel parameters for DB2 data server
<code>vm.swappiness=0</code>	This parameter defines how prone the kernel is to swapping application memory out of physical random access memory (RAM). The default setting, <code>vm.swappiness=0</code> , configures the kernel to give preference to keeping application memory in RAM instead of assigning more memory for file caching. This setting avoids unnecessary paging and excessive use of swap space. It is especially important for data servers configured to use the self-tuning memory manager (STMM).
<code>vm.overcommit_memory=0</code>	This parameter influences how much virtual memory the kernel permits to allocate. The default setting, <code>vm.overcommit_memory=0</code> , sets the kernel to disallow individual processes from making excessively large allocations. However, the total allocated virtual memory is unlimited. Having unlimited virtual memory is important for DB2 data servers, which retain additional unused virtual memory allocations for dynamic memory management. Unreferenced allocated memory is not backed by RAM or paging space on Linux systems. Avoid setting <code>vm.overcommit_memory=2</code> , because this setting limits the total amount of virtual memory that can be allocated, which can result in unexpected errors.

For more information about kernel parameter recommendations, see:

<http://www.ibm.com/developerworks/linux/linux390/perf/>

## 3.3 Large page support

This section describes the `database_memory` parameter as it is used for configuring the database shared memory in order to make use of DB2's large page support. The parameter `DB2_LARGE_PAGE_MEM` is then used to configure support for large page memory for the database shared memory region <sup>4</sup>.

### 3.3.1 Database shared memory size configuration parameter

DB2 has a memory-tuning feature that started in version 9. This self-tuning memory is discussed more in 3.4.1, “Memory and paging considerations for live guest relocation” on page 19. The `database_memory` configuration parameter defines the memory limits used by the tuner. This parameter specifies the amount of memory that is reserved for the database shared memory region.

Setting this parameter to `AUTOMATIC` enables self-tuning. When enabled, the memory tuner determines the overall memory requirements for the database and increases or decreases the amount of memory allocated for database shared memory depending on the current database requirements. For example, if the current database requirements are high, and there is sufficient free memory in the system, more memory will be consumed by database shared memory. Once the database memory requirements drop, or the amount of free memory in the system drops too low, some database shared memory is released.

For environments that do not support the `AUTOMATIC` setting, this should be set to `COMPUTED`. For example, the additional memory can be used for creating new buffer pools, or for increasing the size of existing buffer pools.

To see the results after changing this parameter, see “User-defined database memory” on page 36.

For more information about this parameter, see:

<http://pic.dhe.ibm.com/infocenter/db2luw/v10r1/index.jsp?topic=%2Fcom.ibm.db2.luw.admin.config.doc%2Fdoc%2Fr0006017.html>

### 3.3.2 Enabling large page support

In DB2 10.1 LUW, large page support is available. The `DB2_LARGE_PAGE_MEM` registry variable is used to enable large page support. Setting `DB2_LARGE_PAGE_MEM=DB` enables large-page memory for the database shared memory region, and if `database_memory` is set to `AUTOMATIC`, disables automatic tuning of this shared memory region by `STMM`.

Memory-intensive applications that use large amounts of virtual memory gain performance by using large pages. To enable the DB2 database system to use them, first configure the operating system to use large pages.

The use of large pages requires the `libcap` library. If this option is turned on and the library is not on the system, the DB2 database disables the large kernel pages and continues to function as it would without them.

To verify availability of large pages, known as hugepages on Linux, issue the following command:

```
cat /proc/meminfo
```

If large pages are available, the three lines shown in Example 3-4 on page 18 should display (with different numbers, depending on the amount of memory configured on your server).

---

<sup>4</sup> See the website for up-to-date information, under “Performance variables”:  
<http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/>

#### *Example 3-4 Checking hugepage information*

---

```
HugePages_Total: 200
HugePages_Free: 200
Hugepagesize: 2048 KB
```

---

If you do not see these lines, or if the HugePages\_Total is 0, you need to configure the operating system or kernel. Note that Red Hat Enterprise Linux's default hugepage size is 2048 KB, while SuSE Linux's default is 1024 KB.

### 3.3.3 Testing relocation while DB2 is using large pages

In our test, we put some load on DB2 to force it to use as many large pages as possible. The Linux guest was relocated during that process to several SSI members, and DB2 showed no disruptions. Example 3-5 shows the memory usage, including huge pages.

#### *Example 3-5 Memory information*

---

```
[root@itsolnx4 ~]# cat /proc/meminfo
MemTotal:      20588060 kB
MemFree:       1503464 kB
Buffers:        196 kB
Cached:        19332 kB
SwapCached:    2256 kB
Active:        12016 kB
Inactive:      9548 kB
HighTotal:      0 kB
HighFree:       0 kB
LowTotal:      20588060 kB
LowFree:       1503464 kB
SwapTotal:     719896 kB
SwapFree:      561300 kB
Dirty:         0 kB
Writeback:     0 kB
AnonPages:     1988 kB
Mapped:        8084 kB
Slab:          11100 kB
PageTables:    41468 kB
NFS_Unstable:  0 kB
Bounce:        0 kB
CommitLimit:   1919148 kB
Committed_AS:  2196092 kB
VmallocTotal:  4273987584 kB
VmallocUsed:    4448 kB
VmallocChunk:  4273982832 kB
HugePages_Total: 9272
HugePages_Free: 9206
HugePages_Rsvd: 165
Hugepagesize: 2048 kB
```

---

## 3.4 z/VM memory sizing considerations

The Linux memory model has profound implications for Linux guests running under z/VM<sup>5</sup>:

- ▶ z/VM memory is a shared resource.

Although aggressive caching reduces the likelihood of disk I/O in favor of memory access, you need to consider caching costs. Cached pages in a Linux guest reduce the number of z/VM pages available to other z/VM guests.

- ▶ A large virtual memory space requires more kernel memory.

A larger virtual memory address space requires more kernel memory for Linux memory management. When sizing the memory requirements for a Linux guest, choose the smallest memory footprint that has a minimal effect on the performance of that guest. To reduce the penalty of occasional swapping that might occur in a smaller virtual machine, use fast swap devices (more details in the Redbooks publication *Linux on IBM System z - Performance Measurement and Tuning*).

A 512 MB server does not require all of the memory, but will eventually appear to use all memory because its memory cost is four times that of the 128 MB server.

### 3.4.1 Memory and paging considerations for live guest relocation

In order to test how a z/VM v6.2 live guest relocation would impact memory management, we performed a live guest relocation test while DB2 was under load and using large pages.

The live guest relocation process checks that memory and paging on the destination member is adequate before a relocation of the Linux guest is executed. One memory size check must be finished with a positive result before the relocation can be executed. The current memory size of the guest must fit into the available space of the destination member (Figure 3-1).

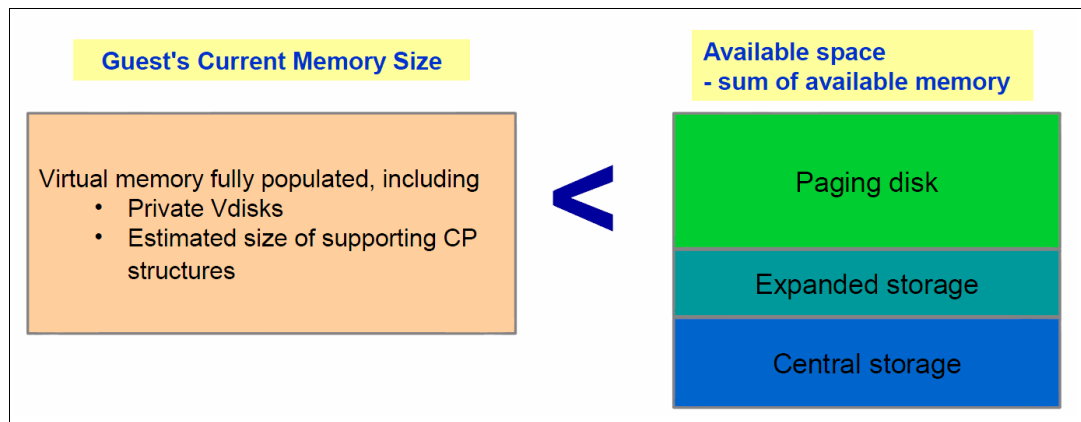


Figure 3-1 Guest storage must fit into available space on the destination member


You must perform additional checks before the relocation starts. Nevertheless, certain checks can be overruled by using the FORCE STORAGE operand with the VMRELOCATE command. For more information about checking for an impact of memory and paging on live guest relocation, see Chapter 3 in *An Introduction to z/VM Single System Image (SSI) and Live Guest Relocation (LGR)*, SG24-8006.

<sup>5</sup> See *z/VM and Linux Operations for z/OS System Programmers*, SG24-7603 for details

Once we satisfied these requirements, the live guest relocation of our DB2 10 Linux on a System z guest relocated with no problems, even when under a heavy load (4000 users running 2000 transactions for 100,000 iterations).

4.3.6, “Relocating a Linux guest to a memory-constrained cluster member” on page 32 details the live guest relocation.





## **DB2 Self-tuning memory manager and z/VM live guest relocation**

In this chapter we discuss the IBM DB2 Self-Tuning Memory Manager (STMM) and how it performs during live guest relocation. We discuss the different scenarios that we ran with STMM enabled during live guest relocation.

## 4.1 DB2 Self-tuning memory manager

Self-Tuning Memory Manager (STMM) is a lifeline for database administrators having at least one of the following scenarios:

- ▶ A database server that shares a server with other servers
- ▶ Fluctuations in resource allocations in a highly virtualized environment
- ▶ Mixed workloads in a single database
- ▶ Changing workloads over time (that is, daytime transaction and night time batch)
- ▶ Too many databases in the enterprise to be able to tune each one
- ▶ Ad-hoc query workloads

STMM is a single knob for memory management inside the DB2 V10 engine. It allows the DB2 instance itself to set and adjust all of the database shared memory (buffer pools, sort heap, lock list, package cache, catalog cache, and so forth) and set them to the value that will improve your overall performance. When enabled, the memory tuner dynamically distributes available memory resources among the following memory consumers: buffer pools, locking memory, package cache, and sort memory. Self-tuning memory is enabled through the `self_tuning_mem` database configuration parameter

The following memory-related database configuration parameters can be automatically tuned:

- ▶ `database_memory` - Database shared memory size
- ▶ `locklist` - Maximum storage for lock list
- ▶ `maxlocks` - Maximum percent of lock list before escalation
- ▶ `pckcachesz` - Package cache size
- ▶ `sheapthres_shr` - Sort heap threshold for shared sorts
- ▶ `sortheap` - Sort heap size

The tuner works within the memory limits defined by the `database_memory` configuration parameter. The value of `database_memory` can itself be automatically tuned. When self-tuning is enabled for `database_memory` (when it is set to `AUTOMATIC`), the tuner determines the overall memory requirements for the database and increases or decreases the amount of memory allocated for database shared memory, depending on the current database requirements.

## 4.2 Determining DB2 memory configuration

Before discussing various test scenarios using STMM and z/VM live guest relocation, we look at testing database memory configuration settings for the running instance and then the database. Database manager configuration parameters are stored in a file named `db2system`. Database configuration parameters are stored in a file named `SQLDBCON`. These files cannot be directly edited, and can only be changed or viewed via a supplied application programming interface (API) or by a tool that calls that API.

### 4.2.1 Fetching instance memory configuration

There are various ways to fetch DB2 configuration details. We have used both the `db2 get cfg` and the `db2pd` command to gather the information. Example 4-1 shows the output from the command `db2pd -dbmcfg`.

#### Example 4-1 Instance configuration parameters

```
db2inst1@itsodbl:~> db2pd -dbmcfg
```

Database Member 0 -- Active -- Up 0 days 00:08:54 -- Date 06/15/2012 01:34:56

##### Database Manager Configuration Settings:

Description	Memory Value	Disk Value
RELEASE	0xf00	0xf00
CPUSPEED(millisec/instruction)	1.928739e-07	1.928739e-07
COMM_BANDWIDTH(MB/sec)	1.000000e+02	1.000000e+02
NUMDB	32	32
NUMDB_INT	NEEDS RECOMPUTE(32)	NEEDS RECOMPUTE(32)
FEDERATED	NO	NO

DFTDBPATH (memory)	/home/db2inst1	
DFTDBPATH (disk)	/home/db2inst1	
MON_HEAP_SZ (4KB)	AUTOMATIC(90)	AUTOMATIC(90)
JAVA_HEAP_SZ (4KB)	2048	2048
AUDIT_BUF_SZ (4KB)	0	0
<b>INSTANCE_MEMORY</b> (4KB)	AUTOMATIC(1262425)	AUTOMATIC(1262425)
RSTRT_LIGHT_MEM (4KB)	AUTOMATIC(10)	AUTOMATIC(10)
RSTRT_LIGHT_MEM_INT (4KB)	NEEDS RECOMPUTE(0)	NEEDS RECOMPUTE(0)
BACKBUFSZ (4KB)	1024	1024
RESTBUFSZ (4KB)	1024	1024
SHEAPTHRES (4KB)	0	0
DIR_CACHE	YES	YES
ASLHEAPSZ (4KB)	15	15
RQRIOBLK (bytes)	32767	32767
UTIL_IMPACT_LIM	10	10
AGENTPRI	SYSTEM	SYSTEM

## 4.2.2 Fetching database memory configuration

Before discussing various test scenarios for STMM and z/VM live guest relocation, we look at the test database configuration settings shown in Example 4-2.

#### Example 4-2 Sample Database configuration

```
db2inst1@itsodbl:~> db2pd -dbcfg -db sample
```

Database Member 0 -- Database SAMPLE -- Active -- Up 0 days 13:58:21 -- Date 06/15/2012 00:49:02

##### Database Configuration Settings:

Description	Memory Value	Disk Value
DB configuration release level	0x0F00	0x0F00
Database release level	0x0F00	0x0F00
Database territory	US	US

<b>SELF_TUNING_MEM</b>	OFF	OFF
ENABLE_XMLCHAR	YES	YES
WLM_COLLECT_INT (mins)	0	0

<b>DATABASE_MEMORY</b> (4KB)	AUTOMATIC(43824)	AUTOMATIC(43824)
DB_MEM_THRESH	10	10
LOCKLIST (4KB)	4096	4096
MAXLOCKS	10	10
PCKCACHESZ (4KB)	1200	1200
<b>SHEAPTHRES_SHR</b> (4KB)	5000	5000
<b>SORTHEAP</b> (4KB)	256	256
DBHEAP (4KB)	AUTOMATIC(1200)	AUTOMATIC(1200)
CATALOGCACHE_SZ (4KB)	750	750
LOGBUFSZ (4KB)	256	256
UTIL_HEAP_SZ (4KB)	5000	5000
BUFFPAGE	1000	1000
APPL_MEMORY (4KB)	AUTOMATIC(40016)	AUTOMATIC(40000)
STMHEAP (4KB)	AUTOMATIC(8192)	AUTOMATIC(8192)
APPLHEAPSZ (4KB)	AUTOMATIC(256)	AUTOMATIC(256)
STAT_HEAP_SZ (4KB)	AUTOMATIC(4384)	AUTOMATIC(4384)

From the command listing in Example 4-2, we see that **SELF\_TUNING\_MEM** (STM) is not enabled and the **DATABASE\_MEMORY** allocation is set to automatic. Also take note of the various other parameters such as **SHEAPTHRES\_SHR** and **SORTHEAP**.

**Note:** In older versions of the Linux kernel (prior to RHEL 5 and SLES 10 SP1) the self-tuning memory manager does not allow setting **DATABASE\_MEMORY** to **AUTOMATIC**. However, this setting is now allowed on these kernels only when **INSTANCE\_MEMORY** is set to a specific value, and not **AUTOMATIC**. If **DATABASE\_MEMORY** is set to **AUTOMATIC**, and **INSTANCE\_MEMORY** is later set back to **AUTOMATIC**, the **DATABASE\_MEMORY** configuration parameter is automatically updated to **COMPUTED** during the next database activation.

### 4.2.3 Enabling the self-tuning memory manager

Enable self-tuning memory for the database by setting the `self_tuning_mem` database configuration parameter to **ON** using the `UPDATE DATABASE CONFIGURATION` command as shown in Example 4-3. When enabled, the memory tuner dynamically distributes available memory resources between several memory consumers, including buffer pools, locking memory, package cache, and sort memory.

*Example 4-3 Enablement of STMM*

```
db2inst1@itsodbl:~> db2 update db cfg using SELF_TUNING_MEM ON
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
```

**Note:** For the self-tuning memory manager to be active there must be at least two memory consumers enabled because the memory tuner trades memory resources between different memory consumers.

### 4.2.4 Verification of enablement

To see whether the self-tuning memory manager has been activated, check the database configuration parameters, as shown in Example 4-4.

*Example 4-4 STM enablement verification*

```
db2inst1@itsodbl:~> db2pd -dbcfg -db sample
```

Database Member 0 -- Database SAMPLE -- Active -- Up 0 days 00:00:02 -- Date 06/15/2012 01:40:31

Database Configuration Settings:

Description	Memory Value	Disk Value
DB configuration release level	0x0F00	0x0F00
Database release level	0x0F00	0x0F00
Database territory	US	US
Database code page	1208	1208
Multi-page File alloc enabled	YES	YES
<b>SELF_TUNING_MEM</b>	<b>ON</b>	<b>ON</b>
ENABLE_XMLCHAR	YES	YES
WLM_COLLECT_INT (mins)	0	0
DATABASE_MEMORY (4KB)	AUTOMATIC(43824)	AUTOMATIC(43824)
DB_MEM_THRESH	10	10
LOCKLIST (4KB)	4096	4096
MAXLOCKS	10	10
PCKCACHESZ (4KB)	1200	1200
SHEAPTHRES_SHR (4KB)	5000	5000
SORTHEAP (4KB)	256	256
DBHEAP (4KB)	AUTOMATIC(1200)	AUTOMATIC(1200)
CATALOGCACHE_SZ (4KB)	750	750
LOGBUFSZ (4KB)	256	256
UTIL_HEAP_SZ (4KB)	5000	5000
BUFFPAGE	1000	1000
APPL_MEMORY (4KB)	AUTOMATIC(40016)	AUTOMATIC(40000)
STMHEAP (4KB)	AUTOMATIC(8192)	AUTOMATIC(8192)
APPLHEAPSZ (4KB)	AUTOMATIC(256)	AUTOMATIC(256)
STAT_HEAP_SZ (4KB)	AUTOMATIC(4384)	AUTOMATIC(4384)

**Important:** On Linux operating systems, although the **ipcs** command can be used to list all the shared memory segments, it does not accurately reflect the amount of resources consumed. You can use the **db2mtrk** command as an alternative to **ipcs**.

## 4.3 DB2 LGR on a memory-constrained target cluster member

We have taken ITSOSI3 and ITSOSI4 z/VM members and configured them with 30 GB and 8 GB of real memory, respectively. We relocated the DB2 Linux on the System z guest from ITSOSI3 (the member with more memory resources) to ITSOSI4 where there is a memory constraint. Also, to make the environment production-like, we have other guests loaded on to the ITSOSI4 member with their respective workloads running. The z/VM environments for STMM are shown in Figure 4-1 on page 26.

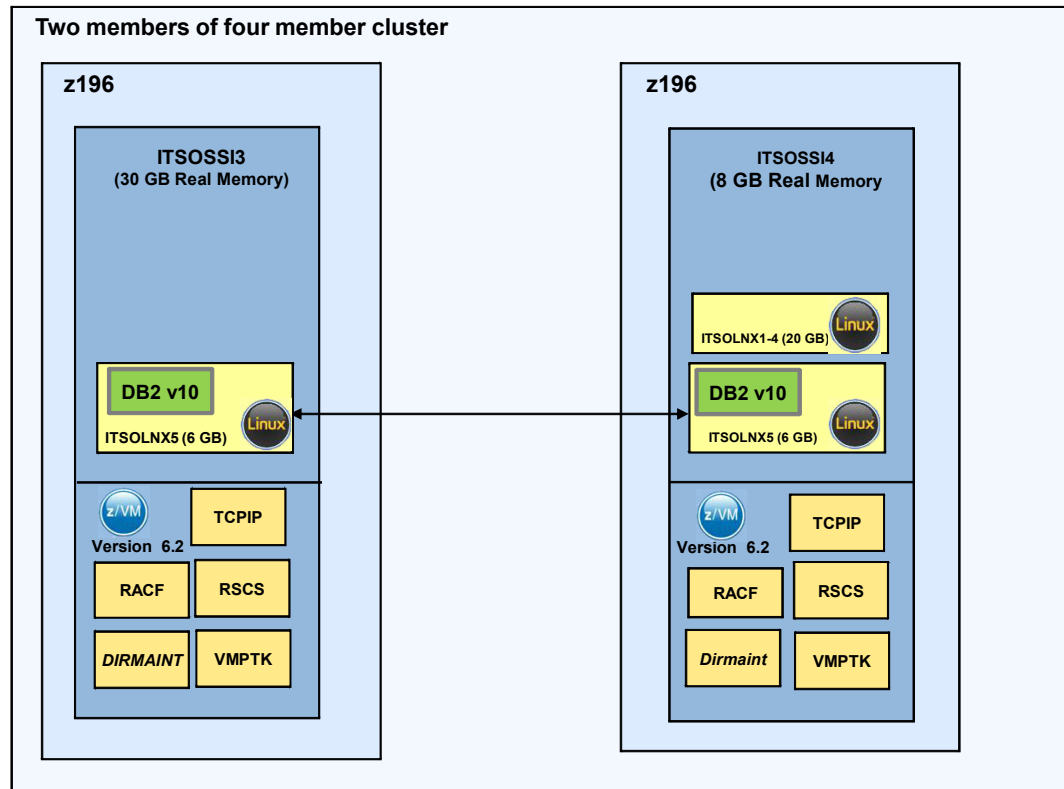


Figure 4-1 z/VM member environment for STMM

### ITSOSI4 member

Example 4-5 shows the details for the SSI cluster named ITSOSIA and shows members of that cluster, ITSOSI1-ITSOSI5.

#### Example 4-5 Cluster details

```
q ssi
SSI Name: ITSOSIA
SSI Mode: Stable
Cross-System Timeouts: Enabled
SSI Persistent Data Record (PDR) device: SSIAC2 on 9E20
SLOT SYSTEMID STATE      PDR HEARTBEAT      RECEIVED HEARTBEAT
  1 ITSOSI1 Joined    06/19/12  14:24:45 06/19/12  13:19:06
  2 ITSOSI2 Joined    06/19/12  13:19:05 06/19/12  13:19:05
  3 ITSOSI3 Joined    06/19/12  13:19:07 06/19/12  13:19:07
  4 ITSOSI4 Joined    06/19/12  13:19:31 06/19/12  13:19:31
Ready; T=0.01/0.01 13:19:34

q stor
STORAGE = 8G CONFIGURED = 8G INC = 256M STANDBY = 0  RESERVED = 0
Ready; T=0.01/0.01 13:19:53
```

### ITSOSI3 member

Example 4-6 on page 27 shows the storage details for cluster member ITSOSI3. Note that it includes ITSOSI3 in its cluster details.

Example 4-6 Details of member ITSOSI3

```
q stor
STORAGE = 30G CONFIGURED = 30G INC = 256M STANDBY = 0 RESERVED = 0
Ready; T=0.01/0.01 13:22:30
```

For simulating a workload, we created a table (named LOADTAB) under the sample database, and populated it with approximately 60,000 records. These records will be sorted and selected using a couple of complex SQL queries.

We enabled the DB2 sort heap and buffer pools for self-tuning to demonstrate how the sort heap and buffer pool pages are tuned automatically by DB2's self-tuning memory over time. From the database memory configuration, we used the sort heap and buffer pool parameters for the tests.

The sort heap (SORTHEAP) determines the maximum number of memory pages that can be used for each sort. The SORTHEAP parameter is used by the optimizer to determine whether the sorting can be performed in memory or on disk. DB2 always attempts to perform the sort in memory. Likewise, buffer pools are database objects used to cache data pages in memory. Once a data page is placed in a buffer pool, physical I/O access to disk can be avoided. Buffer pools can be assigned to cache only a particular table space. Buffer pools are a very important tuning area in DB2 performance tuning.

During the execution of the test we tracked the memory allocations and the adjustments done by the self-tuning memory manager and compared to see how STM identifies the memory constraint and adjusts sort heap size and buffer pools for optimal performance.

### 4.3.1 Verifying database sort heap size and bufferpool page size

First we started with checking the current values of the database parameters SELF\_TUNING\_MEM, SHEAPTHRES\_SHR, and SORTHEAP, as shown in Example 4-7.

Example 4-7 Current SORTHEAP and Buffer Pool size

```
db2inst1@itsodbl:~> db2 get db cfg for sample show detail
```

Database Configuration for Database sample

Description	Parameter	Current Value
Database configuration release level		= 0x0f00
Database release level		= 0x0f00
Database territory		= US
<b>Self tuning memory</b>	<b>(SELF_TUNING_MEM)</b>	<b>= ON (Active)</b>
Size of database shared memory (4KB)	(DATABASE_MEMORY)	= AUTOMATIC(51824)
Database memory threshold	(DB_MEM_THRESH)	= 10
Max storage for lock list (4KB)	(LOCKLIST)	= AUTOMATIC(10368)
Percent. of lock lists per application	(MAXLOCKS)	= AUTOMATIC(98)
Package cache size (4KB)	(PCKCACHESZ)	= (MAXAPPLS*8)
<b>Sort heap thres for shared sorts (4KB)</b>	<b>(SHEAPTHRES_SHR)</b>	<b>= 5000</b>
<b>Sort list heap (4KB)</b>	<b>(SORTHEAP)</b>	<b>= 256</b>

```
db2inst1@itsodbl:~> db2 get snapshot for bufferpools on sample
```

#### Bufferpool Snapshot

Bufferpool name	= IBMDEFAULTBP
Database name	= SAMPLE
Database path	= /home/db2inst1/db2inst1/NODE0000/SQL0
Input database alias	= SAMPLE
Snapshot timestamp	= 06/18/2012 22:51:50.152480
Node number	= 0
Tablespaces using bufferpool	= 6
Alter bufferpool information:	
Pages left to remove	= 0
<b>Current size</b>	<b>= 1000</b>
<b>Post-alter size</b>	<b>= 1000</b>

---

### 4.3.2 Varying database sort heap and buffer pool size

From the information previously gathered, sort heap and buffer pools are currently configured to be 256 and 1000 pages respectively. Enabling the sort heap and buffer pools for self-tuning and running a workload demonstrates how the sort heap and buffer pools are tuned automatically by the DB2 self-tuning memory manager over time. Example 4-8 shows the commands used to update these parameters.

#### *Example 4-8 Setting SORT HEAP and Buffer Pool new page size*

---

```
db2inst1@itsodbl:~> db2 connect to sample
```

#### Database Connection Information

Database server	= DB2/LINUXZ64 10.1.0
SQL authorization ID	= DB2INST1
Local database alias	= SAMPLE

```
db2inst1@itsodbl:~> db2 update db cfg using SELF_TUNING_MEM ON
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
```

```
db2inst1@itsodbl:~> db2 update db cfg using SORTHEAP 16 IMMEDIATE
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
```

```
db2inst1@itsodbl:~> db2 alter bufferpool IBMDEFAULTBP IMMEDIATE SIZE 50 AUTOMATIC
DB20000I The SQL command completed successfully.
```

```
db2inst1@itsodbl:~> db2 update monitor switches using lock on
DB20000I The UPDATE MONITOR SWITCHES command completed successfully.
db2inst1@itsodbl:~>
```

---



### 4.3.3 Verifying new database configuration

Before verifying the new database configuration, shut down and restart the DB2 engine. This refreshes the memory heap and the new setting comes into effect. Example 4-9 shows verification of the new database configuration.

*Example 4-9 Verifying new database configuration*

---

```
db2inst1@itsodbl:~> db2pd -dbcfg -db sample
```

```
Database Member 0 -- Database SAMPLE -- Active -- Up 0 days 00:00:07 -- Date
06/18/2012 23:34:04
```

Database Configuration Settings:

Description	Memory Value	Disk Value
DB configuration release level	0x0F00	0x0F00
Database release level	0x0F00	0x0F00
Database territory	US	US
SELF_TUNING_MEM	ON	ON
ENABLE_XMLCHAR	YES	YES
WLM_COLLECT_INT (mins)	0	0
DATABASE_MEMORY (4KB)	AUTOMATIC(49712)	AUTOMATIC(49712)
DB_MEM_THRESH	10	10
LOCKLIST (4KB)	10400	10400
MAXLOCKS	AUTOMATIC(97)	AUTOMATIC(97)
PCKCACHESZ (4KB)	320	320
SHEAPTHRES_SHR (4KB)	5000	5000
<b>SORTHEAP (4KB)</b>	<b>16</b>	<b>16</b>

```
db2inst1@itsodbl:~> db2 get snapshot for bufferpools on sample
```

Bufferpool Snapshot

Bufferpool name	= IBMDEFAULTBP
Database name	= SAMPLE
Database path	=
/home/db2inst1/db2inst1/NODE0000/SQL00002/MEMBER0000/	
Input database alias	= SAMPLE
Snapshot timestamp	= 06/19/2012 00:12:32.104990
Node number	= 0
Tablespaces using bufferpool	= 6
Alter bufferpool information:	
Pages left to remove	= 0
Current size	= 50
Post-alter size	= 50

---

### 4.3.4 Load generation queries

Once the new database configurations for the sort heap and buffer pools are defined, the Self-Tuning Memory Manager can be tested. During the entire cycle of these tests, we used the **db2mtrk** command to record the memory usage.

A database test depends on how the workloads use the various DB2 subsystems, especially memory. For example, queries with many ORDER BY clauses may gain more benefits from greater sort space than from greater bufferpool space. So we used two SQL queries to simulate a complex sort and select operation, which would attempt to use most of the sort heap and the buffer pools allocated respectively. When DB2 self-tuning memory manager finds that there needs to be some adjusting to the memory configuration, it tunes those optimally.

---

*Example 4-10 Starting memory tracking*

---

```
db2inst1@itsodbl:~> db2mtrk -d -v -r 10 > workload_stats.txt &  
[1] 47012
```

---

For the sake of demonstrating how the sort heap and buffer pools are tuned automatically by DB2, we purposely created complex queries to test. In our tests, Query1 (q1.sql in Example 4-11) will do a lot of sort operations and Query2 (q2.sql in Example 4-11) will be a more complex select operation.

---

*Example 4-11 Execution of load generation queries*

---

```
db2inst1@itsodbl:~> db2 -tvf q1.sql
```

```
db2inst1@itsodbl:~> db2 -tvf q2.sql
```

---

Once the SQL queries are finished, stop the DB2 memory tracker (**db2mtrk**) using the command line prompt by pressing the Ctrl + c keys or the **kill** command.

---

*Example 4-12 Stopping db2 memory tracker*

---

```
db2inst1@itsodbl:~> kill -9 47012  
[1]+  Killed                  db2mtrk -d -v -r 10 > workload_stats.txt
```

---

### 4.3.5 Analysis of the DB2 memory tracker report

The results in Example 4-13 clearly show that the self-tuning memory was behaving consistently as per our expectations. During runtime, the self-tuning memory manager dynamically updates the size of performance-critical heaps within the database shared memory set according to the free physical memory on the system, while ensuring that there is sufficient free `instance_memory` available for functional memory requirements.

For this database, STMM was dynamically shifting memory to SortHeap and into the buffer pools to help address the prolific and costly scans and sorts. While 16 pages had been defined in the database configuration, DB2 automatically increased the SortHeap up to 304 pages (1,245,184 bytes) on the system. For the buffer pool size, the initial configuration was set to 50 pages. Being too small for a database of this size, STMM normalized immediately to around 200 pages (983040 bytes). But when we actually started the SQL queries, the buffer pool was adjusted to 2224 pages (9,109,504 bytes).

See Example 4-13 for memory configuration before query execution and Figure 4-14 on page 31 for memory configuration during query execution.

---

*Example 4-13 Memory tracking report from db2mtrk before query execution*

---

```
Tracking Memory on: 2012/06/19 at 10:27:42
```

```
Memory for database: SAMPLE
```

Backup/Restore/Util Heap is of size 65536 bytes  
 Package Cache is of size 1376256 bytes  
 Other Memory is of size 196608 bytes  
 Catalog Cache Heap is of size 458752 bytes  
**Buffer Pool Heap (1) is of size 983040 bytes**  
 Buffer Pool Heap (System 32k buffer pool) is of size 851968 bytes  
 Buffer Pool Heap (System 16k buffer pool) is of size 589824 bytes  
 Buffer Pool Heap (System 8k buffer pool) is of size 458752 bytes  
 Buffer Pool Heap (System 4k buffer pool) is of size 393216 bytes  
**Shared Sort Heap is of size 65536 bytes**  
 Lock Manager Heap is of size 44695552 bytes  
 Database Heap is of size 55181312 bytes  
 Application Heap (27) is of size 131072 bytes  
 Application Heap (24) is of size 65536 bytes  
 Application Heap (23) is of size 65536 bytes  
 Application Heap (22) is of size 65536 bytes  
 Application Heap (21) is of size 196608 bytes  
 Application Heap (20) is of size 65536 bytes  
 Application Heap (19) is of size 65536 bytes  
 Application Heap (18) is of size 131072 bytes  
 Application Heap (17) is of size 131072 bytes  
 Applications Shared Heap is of size 524288 bytes  
 Total: 106758144 bytes

---

*Example 4-14 Memory tracking report from db2mtrk during query execution*

---

Tracking Memory on: 2012/06/19 at 10:27:44

Memory for database: SAMPLE

Backup/Restore/Util Heap is of size 65536 bytes  
 Package Cache is of size 1376256 bytes  
 Other Memory is of size 196608 bytes  
 Catalog Cache Heap is of size 458752 bytes  
**Buffer Pool Heap (1) is of size 9109504 bytes**  
 Buffer Pool Heap (System 32k buffer pool) is of size 851968 bytes  
 Buffer Pool Heap (System 16k buffer pool) is of size 589824 bytes  
 Buffer Pool Heap (System 8k buffer pool) is of size 458752 bytes  
 Buffer Pool Heap (System 4k buffer pool) is of size 393216 bytes  
**Shared Sort Heap is of size 1245184 bytes**  
 Lock Manager Heap is of size 44695552 bytes  
 Database Heap is of size 55181312 bytes  
 Application Heap (27) is of size 131072 bytes  
 Application Heap (24) is of size 65536 bytes  
 Application Heap (23) is of size 65536 bytes  
 Application Heap (22) is of size 65536 bytes  
 Application Heap (21) is of size 196608 bytes  
 Application Heap (20) is of size 65536 bytes  
 Application Heap (19) is of size 65536 bytes  
 Application Heap (18) is of size 131072 bytes  
 Application Heap (17) is of size 131072 bytes  
 Applications Shared Heap is of size 524288 bytes  
 Total: 114884608 bytes

---

### 4.3.6 Relocating a Linux guest to a memory-constrained cluster member

Now that we had results of the self-tuning memory manager adjusting the sort heap and buffer pools page size, we relocated the Linux guest using z/VM v6.2 live guest relocation on a severely over-committed z/VM member. ITS0SSI4 is configured with 8 GB of real memory and hosts four Linux on System z guests in addition to the Linux on System z guest that hosts the DB2 server. The memory is over-committed by almost four times. We simulated a production-like scenario, running the Trade6 application through a workload generator on one of the guests.

When trying to relocate the DB2 guest to the ITS0SSI4 cluster member, the command returned messages showing that the memory was not sufficient on the target cluster.

*Example 4-15 Relocating itsolnx5 guest to ITS0SSI4 z/VM cluster member*

---

```
Ready; T=0.01/0.01 14:44:20
vmrelocate test itsolnx5 itsossi4
Relocation of user ITSOLNX5 from ITS0SSI3 to ITS0SSI4 did not complete. Guest has
not been moved
HCPRLH1940E ITSOLNX5 is not relocatable for the following reason(s):
HCPRL1811I ITSOLNX5: Maximum storage use (6340896K) exceeds available capacity on
destination (5097164K) by 1243732K
HCPRL1813I ITSOLNX5: Maximum pageable storage use (6192M) exceeds available
auxiliary paging space on destination (4581732K) by 1758876K
Ready(01940); T=0.01/0.01 14:44:52
```

---

One of the messages shown was that the target z/VM cluster member did not have sufficient paging space. Adding more paging space avoids the message on pageable storage space. For our test we had anticipated this and went ahead by forcing the guest to relocate using the FORCE STORAGE command. For more information, refer to Appendix 1.8 in *An Introduction to z/VM Single System Image (SSI) and Live Guest Relocation (LGR)*, SG24-8006.

*Example 4-16 Relocation of DB2 guest ITSOLNX5 to ITS0SSI4*

---

```
vmrelocate move itsolnx5 itsossi4 force storage
Relocation of ITSOLNX5 from ITS0SSI3 to ITS0SSI4 started with FORCE STORAGE in
effect
User ITSOLNX5 has been relocated from ITS0SSI3 to ITS0SSI4
Ready; T=0.01/0.01 14:52:07
```

---

### 4.3.7 Verification of z/VM health after the relocation

In the target z/VM cluster member (ITS0SSI4), there were already two guests with active Trade6 workloads and another one with a memory-intensive workload. Since the target z/VM cluster member already had guests consuming all the allocated memory, it was expected that when the DB2 guest was relocated the overall performance of the environment would degrade.

It is a good practice to verify and monitor the z/VM cluster member health after every guest relocation. Since severe memory constraint was anticipated, we watched the z/VM page allocation and the actual memory utilization. The IBM z/VM Performance Toolkit can provide the overall health of the system conforming to memory utilization of the LPAR as well as individual guests.

The z/VM page allocation can be seen in Example 4-17.

*Example 4-17 z/VM Page allocation snapshot.*

---

q alloc page							
VOLID	RDEV	EXTENT START	EXTENT END	TOTAL PAGES	PAGES IN USE	HIGH PAGE	% USED
-----	----	-----	-----	-----	-----	-----	----
SSI4P2	9E2D	1	10016	1761K	708049	1454K	39%
				-----	-----		----
SUMMARY				1761K	708049		39%
USABLE				1761K	708049		39%
Ready; T=0.01/0.01 15:04:07							

---

**Note:** As a general rule, paging space on DASD should not be allowed to eclipse 30% utilization because the available paging space can become very fragmented and can degrade the performance of the overall environment.

Example 4-18 shows the z/VM storage allocation report from the IBM VM Performance Toolkit.

*Example 4-18 Performance Toolkit z/VM storage allocation report*

---

FCX103	CPU 2817	SER B3BD5	Interval 15:02:29 - 15:03:29	Remote Data
Main storage utilization:			XSTORE utilization:	
Total real storage	8'192MB		Total available	2'048MB
Total available	8'192MB		Att. to virt. machines	0kB
Offline storage frames	0		Size of CP partition	2'048MB
SYSGEN storage size	8'192MB		CP XSTORE utilization	100%
Shared storage	7'320kB		Low threshold for migr.	4'480kB
FREE stor. subpools	3'756kB		XSTORE allocation rate	32/s
Subpool stor. utilization	87%		Average age of XSTORE blks	13057s
Total DPA size	8'069MB		Average age at migration	71546s
Locked pages	3300			
Trace table	1'300kB		MDCACHE utilization:	
Pageable	8'054MB		Min. size in XSTORE	0kB
<b>Storage utilization</b>	<b>114%</b>		Max. size in XSTORE	2'048MB
Tasks waiting for a frame	1		Ideal size in XSTORE	97'452kB
Tasks waiting for a page	0/s		Act. size in XSTORE	577'536 B
Standby real stor. size	0kB		Bias for XSTORE	1.00
Reservd real stor. size	0kB		Min. size in main stor.	0kB
			Max. size in main stor.	8'192MB
			Ideal size in main stor.	0kB
Paging / spooling activity:				
<b>FCXVMC111I Critical exception message(s) issued</b>				

---

From the Performance Toolkit report shown in Example 4-18, the storage utilization has peaked to about 114%, which means that the LPAR is already overcommitted on memory. We did not want to resolve the critical exception reported here since the purpose of the test was to find how STMM adjusts the resources during a relocation.

### 4.3.8 Load generation and database memory setup

We followed the same procedure for setting up the database memory configuration and then simulating the workload as in the previous sections. The only exception was that the current

environment was memory-constrained and z/VM was already heavily paging to its paging devices, as shown in Example 4-19.

*Example 4-19 Memory tracker report before query execution*

---

```
Tracking Memory on: 2012/06/20 at 12:08:21
Memory for database: SAMPLE
Backup/Restore/Util Heap is of size 65536 bytes
Package Cache is of size 1376256 bytes
Other Memory is of size 196608 bytes
Catalog Cache Heap is of size 524288 bytes
Buffer Pool Heap (1) is of size 983040 bytes
Buffer Pool Heap (System 32k buffer pool) is of size 851968 bytes
Buffer Pool Heap (System 16k buffer pool) is of size 589824 bytes
Buffer Pool Heap (System 8k buffer pool) is of size 458752 bytes
Buffer Pool Heap (System 4k buffer pool) is of size 393216 bytes
Shared Sort Heap is of size 65536 bytes
Lock Manager Heap is of size 46989312 bytes
Database Heap is of size 55181312 bytes
Application Heap (19) is of size 131072 bytes
Application Heap (14) is of size 65536 bytes
Application Heap (13) is of size 65536 bytes
Application Heap (12) is of size 65536 bytes
Application Heap (11) is of size 196608 bytes
Application Heap (10) is of size 65536 bytes
Application Heap (9) is of size 65536 bytes
Application Heap (8) is of size 131072 bytes
Application Heap (7) is of size 131072 bytes
Applications Shared Heap is of size 655360 bytes
Total: 109248512 bytes
```

---

In Example 4-20, we see that DB2 adjusted the user-defined buffer pool size of 50 pages in two stages. During the first stage, it increased the buffer pool size to 240 pages and then adjusted to 2224 pages. The same happened with the sort heap size, with 48 pages adjusted to 272 pages from the user-defined 16 pages.

*Example 4-20 Memory tracker report during query execution*

---

```
Tracking Memory on: 2012/06/20 at 12:09:16
Memory for database: SAMPLE
Backup/Restore/Util Heap is of size 65536 bytes
Package Cache is of size 1376256 bytes
Other Memory is of size 196608 bytes
Catalog Cache Heap is of size 524288 bytes
Buffer Pool Heap (1) is of size 9109504 bytes
Buffer Pool Heap (System 32k buffer pool) is of size 851968 bytes
Buffer Pool Heap (System 16k buffer pool) is of size 589824 bytes
Buffer Pool Heap (System 8k buffer pool) is of size 458752 bytes
Buffer Pool Heap (System 4k buffer pool) is of size 393216 bytes
Shared Sort Heap is of size 1114112 bytes
Lock Manager Heap is of size 46989312 bytes
Database Heap is of size 55181312 bytes
Application Heap (19) is of size 131072 bytes
Application Heap (14) is of size 65536 bytes
Application Heap (13) is of size 65536 bytes
Application Heap (12) is of size 65536 bytes
```

```

Application Heap (11) is of size 196608 bytes
Application Heap (10) is of size 65536 bytes
Application Heap (9) is of size 65536 bytes
Application Heap (8) is of size 131072 bytes
Application Heap (7) is of size 131072 bytes
Applications Shared Heap is of size 655360 bytes
Total: 118423552 bytes

```

---

Even though DB2's self-tuning memory manager adjusted the sort heap and buffer pool sizes, we observed that the query execution was slow due to extensive memory constraint in the underlying environment. DB2's STMM feature automatically detected changes in memory usage and resized DB2 heaps to help maximize performance.

Reviewing the results and various DB2 database configurations, we found that the query execution consumes more time when database\_memory is defined to be automatic. If database\_memory is set to a specific value (user defined), then that requested amount of memory is allocated initially, during database activation. If database\_memory is set to automatic, the self-tuner discovers free memory on the host system and tries to tune DB2 memory parameters.

To clear any doubt, we ran the same workloads on the memory-constrained z/VM cluster member with database\_memory defined as *automatic* (Example 4-21) and then the next run with a user-defined memory setting with a roughly estimated size of around 100 MB.

---

*Example 4-21 Setting database memory configuration to automatic*

---

```

db2inst1@itsodb1:~> db2 UPDATE DATABASE CONFIGURATION USING DATABASE_MEMORY
AUTOMATIC ;
DB20000I The UPDATE DATABASE CONFIGURATION command completed successfully.
db2inst1@itsodb1:~>

```

```

db2inst1@itsodb1:~> db2pd -dbcfg -db sample

```

SELF_TUNING_MEM	ON	ON
ENABLE_XMLCHAR	YES	YES
WLM_COLLECT_INT (mins)	0	0
<b>DATABASE_MEMORY (4KB)</b>	<b>AUTOMATIC(51808)</b>	<b>AUTOMATIC(51440)</b>
DB_MEM_THRESH	10	10
LOCKLIST (4KB)	10368	10368
MAXLOCKS	AUTOMATIC(98)	AUTOMATIC(98)
PCKCACHESZ (4KB)	320	320
SHEAPTHRES_SHR (4KB)	5000	5000
SORTHEAP (4KB)	16	16
DBHEAP (4KB)	AUTOMATIC(1200)	AUTOMATIC(1200)
CATALOGCACHE_SZ (4KB)	200	200
LOGBUFSZ (4KB)	256	256

---

We executed the queries prefixed using the **time** command (Example 4-22) to find the elapsed time on each of these queries. This enabled us to compare the elapsed time of both scenarios.

---

*Example 4-22 Execution of load generation queries*

---

```

db2inst1@itsodb1:~> time db2 -tvf q1.sql

```

```

60175 record(s) selected.

```

```

real    1m47.966s
user    0m0.118s
sys     0m0.286s
db2inst1@itsodbl:~>

```

```

db2inst1@itsodbl:~> time db2 -tvf q2.sql
60175 record(s) selected.

```

```

real    1m46.745s
user    0m0.117s
sys     0m0.284s
db2inst1@itsodbl:~>

```

## User-defined database memory

We then configured the database memory (as discussed in 3.3.1, “Database shared memory size configuration parameter” on page 17) to a user-defined size of 400 MB and followed the same procedure as in the previous steps (Example 4-23).

*Example 4-23 Setting database memory configuration to user-defined*

```

db2inst1@itsodbl:~> db2 UPDATE DATABASE CONFIGURATION USING DATABASE_MEMORY 102400
IMMEDIATE ;
DB20000I  The UPDATE DATABASE CONFIGURATION command completed successfully.
db2inst1@itsodbl:~>

```

```

db2inst1@itsodbl:~> db2pd -dbcfg -db sample

```

SELF_TUNING_MEM	ON	ON
ENABLE_XMLCHAR	YES	YES
WLM_COLLECT_INT (mins)	0	0
<b>DATABASE_MEMORY (4KB)</b>	<b>102400</b>	<b>102400</b>
DB_MEM_THRESH	10	10
LOCKLIST (4KB)	10368	10368
MAXLOCKS	AUTOMATIC(98)	AUTOMATIC(98)
PCKCACHESZ (4KB)	320	320
SHEAPTHRES_SHR (4KB)	5000	5000
SORTHEAP (4KB)	16	16
DBHEAP (4KB)	AUTOMATIC(1200)	AUTOMATIC(1200)

During this test, we found that using the user-defined database\_memory, there was better performance than with automatic configuration (see Example 4-24).

*Example 4-24 Execution of load generation queries*

```

db2inst1@itsodbl:~> time db2 -tvf q1.sql

```

```

60175 record(s) selected.

```

```

real    1m18.391s
user    0m0.118s
sys     0m0.289s

```

```

db2inst1@itsodbl:~> time db2 -tvf q2.sql

```



60175 record(s) selected.

<b>real</b>	<b>1m20.347s</b>
user	0m0.119s
sys	0m0.291s

---

## 4.4 Test conclusions

While documentation for z/VM v6.2 recommends setting up cluster members with the same amount of memory, if the target cluster already had other servers, then one could run into memory constraints when performing live guest relocation to the constrained member.

In our case, this may have been caused by less memory available in z/VM as well as the virtual memory being shared by other guests. In a virtualized environment, instead of disabling STMM, provide user-defined buffer size and database memory and keep STMM active because it can tune other things such as sort heap and locklist, among others, which would optimize performance.





## Time synchronization

Time synchronization is important in an SSI cluster. DB2 is sensitive to timestamp changes. In this chapter we discuss hardware clock configuration and a way to keep it always synchronized on Linux using the Network Time Protocol (NTP). We discuss two possible scenarios using hardware clock configuration and how NTP can avoid time synchronization issues. The end result is that DB2 was not affected at all. To test what could affect DB2, we set up the sequence of events demonstrated in 5.1.2, “Relocating DB2 when using the hardware clock” on page 41.

## 5.1 Hardware clock

By default Linux installations get the time information from the hardware clock. Since our hardware is a z/VM host, the time is updated from it during the z/VM ID login (Linux boot). Be aware of your SSI cluster configuration, because depending on how it is configured it may cause different times being set up on different CPCs (defined in Chapter 2, “Our lab environment” on page 7). This issue can be avoided by using Server Time Protocol (STP) on the z/VM level or NTP on the Linux level. See 5.2, “Network Time Protocol (NTP)” on page 52 for more information.

### 5.1.1 Server time protocol (STP)

When z/VM is IPL'ed, the system time of day (TOD) clock is set. It can be set manually or by the Server Time Protocol (STP) facility. The time cannot be changed once it is set, except for the time zone, unless you re-IPL the system again.

As stated in *z/VM v6R2 Systems Operation*, SC24-6233-02, if your hardware is enabled and configured for Server Time Protocol (STP) and this feature has been enabled in your configuration file (using the STP\_timestamping or STP\_tz features statements), you will not receive any prompts to change the TOD clock at IPL time. Instead, STP is automatically used to initialize the TOD clock with the STP-coordinated TOD value. When the TOD clock is initialized, you will receive message HCP986I: TOD Clock Synchronized via STP. z/VM also exploits the STP facility to generate time stamps for guest and system DASD write I/O operations, allowing these I/O operations to be synchronized with those of other systems. This support allows data used by z/VM and its guests to be replicated asynchronously over long distances by IBM System Storage® z/OS Global Mirror (formerly known as Extended Remote Copy, or XRC).

If there are specific requirements to provide accurate time relative to an external time standard for data-processing applications, then consider using the external time source (ETS) function of STP. You can select a dial out service from the HMC or use an NTP service.

The STP feature is recommended but not required for an SSI cluster. On all of our tests STP was turned off and the z/VMs re-IPL'ed. We discovered a problem when we changed the time and STP was turned off. One z/VM system was IPL'ed with a time that was earlier than the other SSI cluster systems. It caused a PLM006 abend and a reboot on the other three systems. z/VM Development has an open issue with this problem and is working on a solution. We saw no problems with the time being set ahead.

If you issue **query stp** with STP\_timestamping enabled and CP has successfully synchronized with STP, you will receive the response shown in Example 5-1.

*Example 5-1 Response from the query stp command*

---

```
Server Time Protocol synchronization activated for timestamping
```

---

To enable STP you need to link, access and edit the SYSTEM CONFIG file. Issue the z/VM commands described in Example 5-2.

*Example 5-2 Accessing SYSTEM CONFIG file to edit*

---

```
cp link pmaint cf0 cf0 mr
access cf0 z
xedit system config z\
```

---

Using XEDIT, follow these steps:

1. Locate Features in the Features Statement section:

```
locate Features
```

2. Locate Enable:

```
locate Enable
```

3. Add three lines after Enable:

```
add 3
```

4. Fill with:

```
STP_TZ      ,  
STP_timestamping ,  
STP_Timezone ,
```

5. Save the file:

```
file
```

6. If you are prepared to shut down and re-ipl the system, issue:

```
Issue: shutdown reipl
```

**Note:** STP\_timestamping tells CP to enable the STP protocol (if the STP facility is installed) and apply timestamps to all XRC-capable DASD devices.

STP\_timezone, or STP\_tz, tells CP to enable the STP protocol (if the STP facility is Issue: shutdown reipl).

During the boot process Linux starts its own software-managed clock off of the hardware clock. Alternatively, the clock can be set through operator commands. After the boot, the Linux clock runs independently of the hardware clock. Because Linux manages its own software clock, it frequently uses NTP (see 5.2, “Network Time Protocol (NTP)” on page 52) to keep its time synchronized to an external source. If the Linux image (logical partition or guest) has a Local Area Network (LAN) connection to an NTP server, Linux can take advantage of it.

### 5.1.2 Relocating DB2 when using the hardware clock

Two tests were done using the hardware clock. In the first test, different timezone configurations were set up on each of the two CPCs, while in the second test the CPC time was manually changed. Linux and DB2 behavior were analyzed on both.

#### Test 1: Changing timezone configuration

The first test checks how Linux, and consequently DB2, handles relocation between two CPCs in different timezones. ITSOSI1 (z10™) is using the Central Standard Time (CST) and ITSOSI3 (z196) is using Eastern Daylight Time (EDT); check Figure 5-1 on page 42.

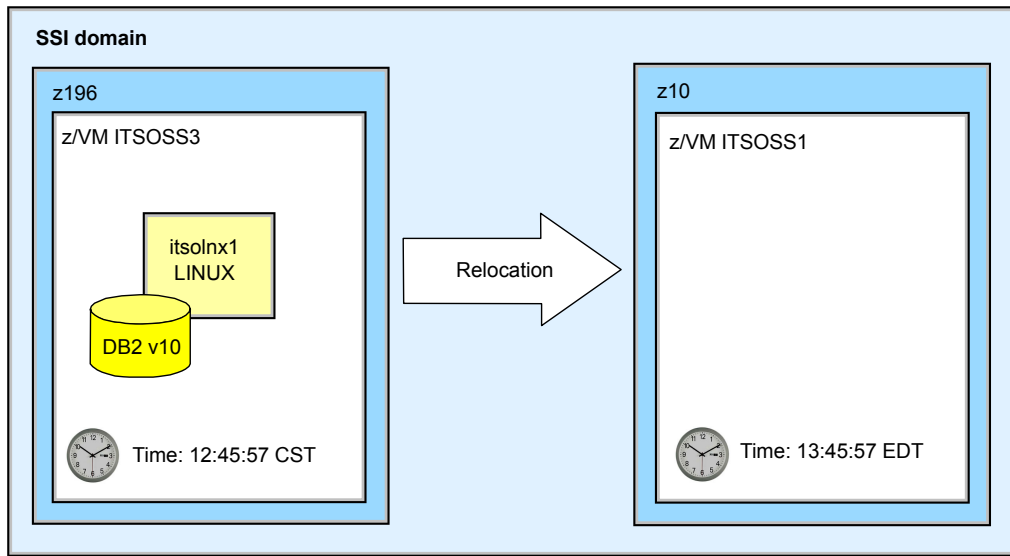


Figure 5-1 CPCs with different timezone configurations

Make sure the *vmcp* module is loaded so we can see on which VM system the server is located by issuing the command shown in Example 5-3.

*Example 5-3 Loading vmcp module*

```
# modprobe vmcp
```

First we have the server set to use the hardware clock. Therefore, the Network Time Protocol (NTP) daemon is not started. NTP is covered in the next section. At this point you simply notice that it is not running during our first test (Example 5-4).

*Example 5-4 Checking NTP daemon*

```
itsolnx1:~ # /etc/init.d/ntp status
Checking for network time protocol daemon (NTPD): unused
```

The Linux guest is on the z/VM member ITS0SSI3 and has the CST timezone; Example 5-5.

*Example 5-5 Checking time settings on Linux and z/VM ITS0SSI3*

```
itsolnx1:~ # vmcp q userid; vmcp q time; date
ITSOLNX1 AT ITS0SSI3
TIME IS 13:45:57 CST MONDAY 06/11/12
CONNECT= 00:01:34 VIRTCPU= 000:04.45 TOTCPU= 000:04.68
Mon Jun 11 13:45:57 CST 2012
```

In Example 5-6, ITS0SSI1 is set to a different timezone, EST (1 hour behind).

*Example 5-6 Checking time settings on ITS0SSI1*

```
q time
TIME IS 12:45:57 EST MONDAY 06/11/12
```

CONNECT= 03:43:09 VIRTCPU= 000:00.00 TOTCPU= 000:00.01

CP READ ITS0SSI1

---

Now, relocating the server (Example 5-7).

*Example 5-7 Relocating from ITS0SSI3 to ITS0SSI1*

---

```
vmrelocate move ITSOLNX1 ITS0SSI1
Relocation of ITSOLNX1 from ITS0SSI3 to ITS0SSI1 started
User ITSOLNX1 has been relocated from ITS0SSI3 to ITS0SSI1
USER DSC LOGOFF AS ITSOLNX1 USERS = 21 FORCED BY SYSTEM
Ready; T=0.01/0.01 13:59:25
```

RUNNING ITS0SSI3

---

Example 5-8 shows fetching date and time again.

*Example 5-8 After relocation the time is not affected*

---

```
itsolnx1:~ # vmcp q userid;vmcp q time;date
ITSOLNX1 AT ITS0SSI1
TIME IS 12:59:31 EST MONDAY 06/11/12
CONNECT= 00:15:08 VIRTCPU= 001:15.94 TOTCPU= 001:16.91
Mon Jun 11 13:59:31 EDT 2012
```

---

Even after a z/VM ID logoff and logon, Example 5-9.

*Example 5-9 Displaying the time after a logoff/logon*

---

```
itsolnx1:~ # uptime; vmcp q userid; date; vmcp q time
2:07pm up 0:01, 1 user, load average: 0.06, 0.03, 0.00
ITSOLNX1 AT ITS0SSI1
Mon Jun 11 14:07:10 EDT 2012
TIME IS 13:07:10 EST MONDAY 06/11/12
CONNECT= 00:01:49 VIRTCPU= 000:05.85 TOTCPU= 000:06.12
```

---

## Results

Linux is smart enough to adjust its timezone just by using the hardware clock. It does not matter if z/VM is set to a different timezone, Linux will always fetch the real time from z/VM and adjust accordingly to its timezone configuration.

## Test 2: Changing the time manually during IPL

In the second test, instead of changing the timezone, we changed the time of one SSI cluster member by 30 minutes to the past during IPL (Example 5-10 on page 44). The Linux guest is not affected by the time change after the relocation, unless you log off the guest. If you log off and log back onto the guest, Linux does not know that the time is wrong (it simply takes the information from the hardware clock) and is not able to fix it since NTP is not enabled. We observed how the cluster, Linux guest, and DB2 server behaved on a relocation from an SSI member with the current time to an SSI member with a past time set. To test what happens to DB2, we set up this sequence of events:

1. STP has been disabled on both SSI cluster members through the system configuration file (Example 5-2 on page 40).
2. We changed the time on this SSI cluster member to 30 minutes in the past to simulate a time synchronization problem between the SSI cluster members.

3. An abend occurred and all members automatically re-IPL'ed (5.1.1, "Server time protocol (STP)" on page 40).
4. We added data to a DB2 database (Example 5-14 on page 46).
5. We took a backup, tested a restore (Example 5-15 on page 46) and relocated to the past time SSI cluster member (Example 5-18 on page 48).
6. To force a time synchronization problem on Linux we logged off and logged back onto the guest.
7. Next we added more data to the DB2 database that was residing on the machine whose time is in the past.
8. We took a new backup, restored the backup file from the machine with the current time, and ran a **roll forward** command.

**Attention:** Although DB2 is not affected during the SSI guest relocation, if the guest is logged off from the SSI member after the relocation, the time will reflect the new SSI member.

This change has been made just to illustrate our test. At the time of this writing, changing the time backwards on one member of the cluster after an IPL causes all members of the cluster to abend and reIPL. Do *not* do it unless it is really necessary. We saw no problems with the time set ahead.

#### *Example 5-10 Setting up z/VM time 30 minutes back*

```
=====?
14:39:08 z/VM V6 R2.0 SERVICE LEVEL 1101 (64-BIT)
14:39:08 SYSTEM NUCLEUS CREATED ON 2012-04-03 AT 15:49:46, LOADED FROM SSI1I2
14:39:08
14:39:08 *****
14:39:08 * LICENSED MATERIALS - PROPERTY OF IBM* *
14:39:08 * * *
14:39:08 * 5741-A07 (C) COPYRIGHT IBM CORP. 1983, 2011. ALL RIGHTS *
14:39:08 * RESERVED. US GOVERNMENT USERS RESTRICTED RIGHTS - USE, *
14:39:08 * DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP SCHEDULE *
14:39:08 * CONTRACT WITH IBM CORP. *
14:39:08 * * *
14:39:08 * * TRADEMARK OF INTERNATIONAL BUSINESS MACHINES. *
14:39:08 *****
14:39:08
14:39:08 *****
14:39:08 * IBM z/VM Single System Image Feature is enabled and active.
14:39:08 *****
14:39:08
14:39:08 HCPZC06718I Using parm disk 1 on volume SSIAC2 (device 9E20).
HCPZC06718I Parm disk resides on cylinders 1 through 120.
14:39:08
14:39:08 HCPZPM6700E File SYSTEM CONFIG, record 197:
14:39:08 HCPZPM6711E Duplicate volume identifier LX6032 specified -- statement
ignored.
14:39:08 HCPMLM3016I Management by the Unified Resource Manager is not available
for this system.
14:39:08 Start ((Warm|Force|COLD|CLEAN) (DRain) (DIsable) (NODIRect)
(NOAUTOlog)) or (SHUTDOWN)
```



```

14:40:41
14:40:41 NOW 14:40:41 EDT MONDAY 2012-06-11
14:40:41 Change TOD clock (Yes|No)
14:40:53 YES
14:40:53 Set date MM/DD/YY or MM/DD/YYYY or YYYY-MM-DD (valid years are 1942
through 2041)
14:41:06 06/11/2012
14:41:06 Set time HH:MM:SS
14:41:34 14:10:00
14:41:34 Press "TOD ENABLE SET" key at designated instant.
14:10:00 NOW 14:10:00 CST MONDAY 2012-06-11
14:10:00 Change TOD clock (Yes|No)
14:10:29 NO
=====

```

Now that we have two different z/VMs with different time settings, as shown in Figure 5-2, we will create a new DB2 table on the sample database, add some records to it, create a backup and move the Linux guest from one CPC to another. On both z/VM members, restore and rollforward operations are executed to check how DB2 behaves with different timestamp settings.

**Note:** You may not notice the time difference on the Linux side just after the relocation, but after an ID logoff/logon, it will get the time from the new CPC. We forced this for test purposes.

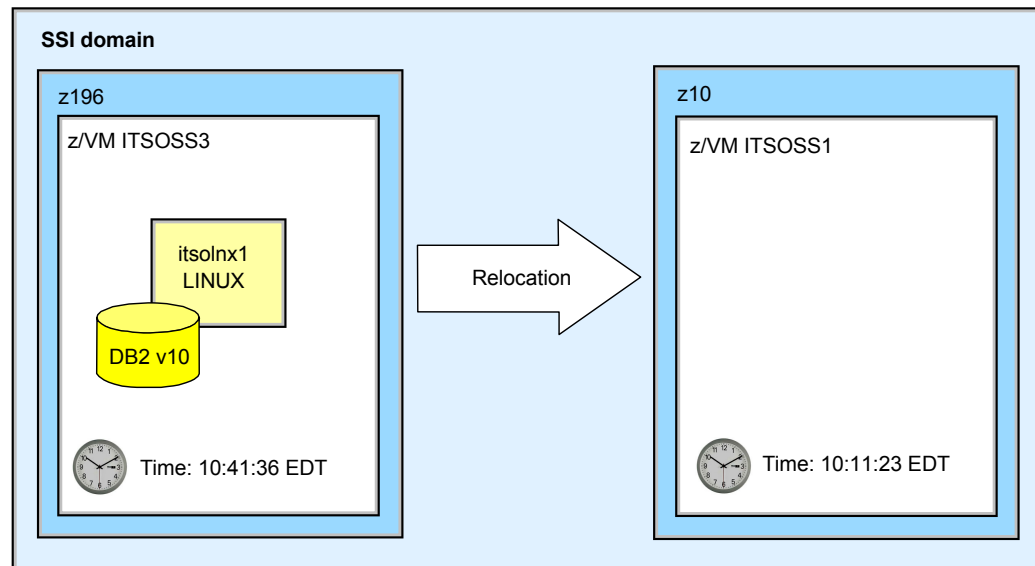


Figure 5-2 CPCs with different time configurations

As you can see in Example 5-11 and Example 5-12, we have two z/VMs with different time settings, thirty minutes apart.

#### Example 5-11 ITS0SSI3 time configuration

```

itsolnx1:~ # vmcp q userid; vmcp q time; date
ITSOLNX1 AT ITS0SSI3
TIME IS 10:41:36 EDT THURSDAY 06/14/12

```

```
CONNECT= 18:51:55 VIRTCPU= 012:50.04 TOTCPU= 013:05.51
Thu Jun 14 10:41:36 EDT 2012
```

---

*Example 5-12 ITSOSI1 time configuration, this one is set to a wrong time*

---

```
q time
TIME IS 10:11:23 EST MONDAY 06/14/12
CONNECT= 03:43:09 VIRTCPU= 000:00.00 TOTCPU= 000:00.01
CP READ ITSOSI1
```

---

On DB2 we created a small table with a timestamp field that uses the default value as the current timestamp (Example 5-13). We used the SAMPLE database that comes with DB2.

*Example 5-13 DB2 table schema used to test timestamp differences*

---

```
CREATE TABLE "DB2INST1"."IN_TRAY2" (
  "RECEIVED" TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  "SOURCE" CHAR(8),
  "SUBJECT" CHAR(64),
  "NOTETEXT" VARCHAR(3000)
)
DATA CAPTURE NONE
IN "USERSPACE1"
COMPRESS NO;
```

---

We added some records to check the timestamp (Example 5-14).

*Example 5-14 First round of inserts using the current timestamp*

---

```
db2inst1@itsolnx1:~> for i in 1 2; do db2 "insert into in_tray2(source,
subject,notetext) VALUES ('right${i}','Subject','Description' )"; done
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
db2inst1@itsolnx1:~>
db2inst1@itsolnx1:~> #checking records...
db2inst1@itsolnx1:~> db2 "select received,source from in_tray2"
```

RECEIVED	SOURCE
2012-06-14-10.41.37.626076	right1
2012-06-14-10.41.37.646395	right2

2 record(s) selected.

---

To check if a rollforward operation was working, we took a backup of the database, inserted more data on the machine that had the current time (Example 5-15) and then tried to restore it (Example 5-16 on page 47).

*Example 5-15 Executing DB2 backup and inserting more data on current time machine*

---

```
db2inst1@itsolnx1:~> db2 backup db sample
```

Backup successful. The timestamp for this backup image is : 20120614104148

```
db2inst1@itsolnx1:~> db2 connect to sample
```

## Database Connection Information

Database server = DB2/LINUXZ64 10.1.0  
SQL authorization ID = DB2INST1  
Local database alias = SAMPLE

```
db2inst1@itsolnx1:~> for i in 11 12; do db2 "insert into in_tray2(source,
subject,notetext) VALUES ('right${i}','Subject','Description' )"; done
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
db2inst1@itsolnx1:~> db2 "select received,source from in_tray2"
```

RECEIVED	SOURCE
2012-06-14-10.41.37.626076	right1
2012-06-14-10.41.37.646395	right2
2012-06-14-10.42.48.478041	right11
2012-06-14-10.42.48.500735	right12

4 record(s) selected.

---

### Example 5-16 Restoring the database and checking the records

---

```
db2inst1@itsolnx1:~> db2 "restore db sample from /home/db2inst1 taken at
20120614104148"
SQL2539W Warning! Restoring to an existing database that is the same as the
backup image database. The database files will be deleted.
Do you want to continue ? (y/n) y
DB20000I The RESTORE DATABASE command completed successfully.

db2inst1@itsolnx1:~> db2 rollforward db sample to end of logs
```

#### Rollforward Status

Input database alias	= sample
Number of members have returned status	= 1
Member ID	= 0
Rollforward status	= DB working
Next log file to be read	= S0000005.LOG
Log files processed	= S0000004.LOG - S0000004.LOG
Last committed transaction	= 2012-06-14-14.42.48.000000 UTC

```
DB20000I The ROLLFORWARD command completed successfully.
db2inst1@itsolnx1:~> db2 rollforward db sample complete
```

#### Rollforward Status

Input database alias	= sample
Number of members have returned status	= 1
Member ID	= 0
Rollforward status	= not pending
Next log file to be read	=

```
Log files processed           = S0000004.LOG - S0000004.LOG
Last committed transaction   = 2012-06-14-14.42.48.000000 UTC
```

DB20000I The ROLLFORWARD command completed successfully.

```
db2inst1@itsolnx1:~> db2 connect to sample
```

Database Connection Information

```
Database server      = DB2/LINUXZ64 10.1.0
SQL authorization ID = DB2INST1
Local database alias = SAMPLE
```

```
db2inst1@itsolnx1:~> db2 "select received,source from in_tray2"
```

RECEIVED	SOURCE
2012-06-14-10.41.37.626076	right1
2012-06-14-10.41.37.646395	right2
2012-06-14-10.42.48.478041	right11
2012-06-14-10.42.48.500735	right12

4 record(s) selected.

---

As expected, everything worked fine. The next step is a relocation of this server from ITS0SSI3 to ITS0SSI1 (Example 5-17) that is configured to 30 minutes earlier and to try to execute the same steps. We need to check if DB2 is able to restore from a backup file from the machine with the current time and what timestamp DB2 used for its data.

*Example 5-17 Relocating Linux to an SSI cluster member that uses time in the past.*

---

```
at itsossi3 cmd vmrelocate move itsolnx1 itsossi1
Relocation of ITSOLNX1 from ITS0SSI3 to ITS0SSI1 started
User ITSOLNX1 has been relocated from ITS0SSI3 to ITS0SSI1
Ready; T=0.01/0.01 10:19:01
```

---

Once the relocation was successful we did an SSI member logoff and logon and picked the new time setting on Linux, as you can see in Example 5-18.

**Note:** We did it just to simulate a power down and to see what could affect DB2 if the guest had been set with a different time. Executing the relocation only, the guest is not affected at all.

*Example 5-18 Linux is now using the ITS0SSI1 with the wrong time configured*

---

```
itsolnx1:~ # vmcp q userid; vmcp q time; date
ITSOLNX1 AT ITS0SSI1
TIME IS 10:25:36 EDT THURSDAY 06/14/12
CONNECT= 00:01:43 VIRTCPU= 000:05.71 TOTCPU= 000:05.97
Thu Jun 14 10:25:36 EDT 2012
```

---

In Example 5-19 we added more records to see the wrong timestamp inserted into the database.

*Example 5-19 Records inserted with a wrong timestamp*

---

```
db2inst1@itsolnx1:~> for i in $(seq 2); do db2 "insert into in_tray2(source,
subject,notetext) VALUES ('wrong${i}','Subject','Description' )"; done
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
db2inst1@itsolnx1:~> db2 "select received,source from in_tray2"
```

RECEIVED	SOURCE
2012-06-14-10.41.37.626076	right1
2012-06-14-10.41.37.646395	right2
2012-06-14-10.42.48.478041	right11
2012-06-14-10.42.48.500735	right12
2012-06-14-10.27.58.757020	wrong1
2012-06-14-10.27.58.784167	wrong2

6 record(s) selected.

---

As you can see in Example 5-19, the data integrity is directly affected by the time change. Records with a time 15 minutes early have been added, not following the logical order.

To test rollforward, we took a backup from the actual state of the database, inserted some more records and tested it by restoring from the backup file that we created while using the current time (Example 5-15 on page 46) and added new records using past time (Example 5-20)

*Example 5-20 New backup from the database after inserting some records*

---

```
db2inst1@itsolnx1:~> db2 backup db sample
```

Backup successful. The timestamp for this backup image is : 20120614104816

```
db2inst1@itsolnx1:~> db2 connect to sample
```

Database Connection Information

Database server	= DB2/LINUXZ64 10.1.0
SQL authorization ID	= DB2INST1
Local database alias	= SAMPLE

---

Adding more data in Example 5-21 to test the rollforward with records using a timestamp in the past.

*Example 5-21 Adding more data to test rollforward*

---

```
db2inst1@itsolnx1:~> for i in $(seq 11 12); do db2 "insert into in_tray2(source,
subject,notetext) VALUES ('wrong${i}','Subject','Description' )"; done
DB20000I The SQL command completed successfully.
DB20000I The SQL command completed successfully.
db2inst1@itsolnx1:~> db2 "select received,source from in_tray2"
```

RECEIVED	SOURCE
----------	--------

```

-----
2012-06-14-10.41.37.626076 right1
2012-06-14-10.41.37.646395 right2
2012-06-14-10.42.48.478041 right11
2012-06-14-10.42.48.500735 right12
2012-06-14-10.27.58.757020 wrong1
2012-06-14-10.27.58.784167 wrong2
2012-06-14-10.30.37.366638 wrong11
2012-06-14-10.30.37.394163 wrong12

```

8 record(s) selected.

---

Using the backup file from Example 5-15 on page 46 we tried to restore the database and run a rollforward to recover all 8 records that we had before; you can see the results in Example 5-22 and Example 5-23 on page 51.

*Example 5-22 Restoring the database using a file from the "future"*

---

```

db2inst1@itsolnx1:~> db2 "restore db sample from /home/db2inst1 taken at
20120614104148"
SQL2539W  Warning!  Restoring to an existing database that is the same as the
backup image database.  The database files will be deleted.
Do you want to continue ? (y/n) y
DB20000I  The RESTORE DATABASE command completed successfully.
db2inst1@itsolnx1:~> db2 rollforward db sample to end of logs

```

Rollforward Status

```

Input database alias           = sample
Number of members have returned status = 1

Member ID                     = 0
Rollforward status            = DB  working
Next log file to be read      = S0000007.LOG
Log files processed            = S0000004.LOG - S0000006.LOG
Last committed transaction    = 2012-06-14-14.42.48.000000 UTC

```

```

DB20000I  The ROLLFORWARD command completed successfully.
db2inst1@itsolnx1:~> db2 rollforward db sample complete

```

Rollforward Status

```

Input database alias           = sample
Number of members have returned status = 1

Member ID                     = 0
Rollforward status            = not pending
Next log file to be read      =
Log files processed            = S0000004.LOG - S0000006.LOG
Last committed transaction    = 2012-06-14-14.42.48.000000 UTC

```

**DB20000I The ROLLFORWARD command completed successfully.**

---

Notice in Example 5-21 on page 49, the time on the last committed record is 10:30:37 and in the rollforward the last committed transaction is in the future (Example 5-22).

*Example 5-23 Checking the records after the restore*

---

```
db2inst1@itsolnx1:~> db2 "select received,source from in_tray2"
```

RECEIVED	SOURCE
2012-06-14-10.41.37.626076	right1
2012-06-14-10.41.37.646395	right2
2012-06-14-10.42.48.478041	right11
2012-06-14-10.42.48.500735	right12
2012-06-14-10.27.58.757020	wrong1
2012-06-14-10.27.58.784167	wrong2
2012-06-14-10.30.37.366638	wrong11
2012-06-14-10.30.37.394163	wrong12

8 record(s) selected.

---

Just to confirm that everything was fine we also restored the backup file from Example 5-20 on page 49 in Example 5-24.

*Example 5-24 Restoring using the other backup file*

---

```
db2inst1@itsolnx1:~> db2 "restore db sample from /home/db2inst1 taken at
20120614104816"
SQL2539W Warning! Restoring to an existing database that is the same as the
backup image database. The database files will be deleted.
Do you want to continue ? (y/n) y
DB20000I The RESTORE DATABASE command completed successfully.
db2inst1@itsolnx1:~> db2 rollforward db sample to end of logs
```

Rollforward Status

Input database alias	= sample
Number of members have returned status	= 1
Member ID	= 0
Rollforward status	= DB working
Next log file to be read	= S0000007.LOG
Log files processed	= S0000006.LOG - S0000006.LOG
Last committed transaction	= 2012-06-14-14.42.48.000000 UTC

```
DB20000I The ROLLFORWARD command completed successfully.
db2inst1@itsolnx1:~> db2 rollforward db sample complete
```

Rollforward Status

Input database alias	= sample
Number of members have returned status	= 1
Member ID	= 0
Rollforward status	= not pending
Next log file to be read	=
Log files processed	= S0000006.LOG - S0000006.LOG

Last committed transaction = 2012-06-14-14.42.48.000000 UTC

DB20000I The ROLLFORWARD command completed successfully.

---

As expected, all records were restored (Example 5-25).

*Example 5-25 All records have been restored successfully even with the wrong timestamp*

---

```
db2inst1@itsolnx1:~> db2 "select received,source from in_tray2"
```

RECEIVED	SOURCE
-----	-----
2012-06-14-10.41.37.626076	right1
2012-06-14-10.41.37.646395	right2
2012-06-14-10.42.48.478041	right11
2012-06-14-10.42.48.500735	right12
2012-06-14-10.27.58.757020	wrong1
2012-06-14-10.27.58.784167	wrong2
2012-06-14-10.30.37.366638	wrong11
2012-06-14-10.30.37.394163	wrong12

8 record(s) selected.

---

### **Results**

As a result of the sequence of events in this case (Example 5-21 and Example 5-25), DB2 restore and rollforward commands are not affected by a configuration using the wrong time. However, the data integrity is affected. Be aware that if your data uses a current timestamp, it picks up the time from the guest and applications may face issues. Avoiding this issue is analyzed in the next section.

## **5.2 Network Time Protocol (NTP)**

A standard way of keeping dates synchronized is by using NTP. It is widely used to synchronize computer clocks via the Internet. It will always keep the time synchronized in your environment, even if it has been relocated to hardware that has the wrong time and the server has been rebooted.

### **5.2.1 Configuring NTP**

The NTP configuration file is located at `/etc/ntp.conf`. If you do not have this file, the NTP package is probably not installed, so you can use the package management tool of your Linux distribution to install it.

A basic configuration file follows in Example 5-26.

*Example 5-26 ntp.conf basic configuration*

---

```
# --- GENERAL CONFIGURATION ---
server aaa.bbb.ccc.ddd

# Drift file.
driftfile /var/lib/ntp/drift/ntp.drift
```

---



This basic configuration defines the server used to synchronize your time and a drift file that stores the error frequency (drift). If the daemon is stopped and restarted it can reinitialize itself to the previous estimate without spending time recomputing the frequency estimate, based on the drift file. This file is required, so please check its permissions to avoid errors.

The file used in our lab had more parameters. This is just an example of the configuration used in our experiments (Example 5-27).

---

*Example 5-27 Configuration file used in our test server*

---

```
itsoln1:~ # cat /etc/ntp.conf
## NTP config file

# path for drift file
driftfile /var/lib/ntp/drift/ntp.drift

# alternate log file
logfile /var/log/ntp
#
# Authentication stuff
#
# path for keys file
keys /etc/ntp.keys
# define trusted keys
trustedkey 1
requestkey 1
server 9.56.248.20 iburst
restrict 9.56.248.20
```

---

This configuration uses some security parameters to control access to the NTP server. When an NTP daemon is first set up, it is also set as an NTP server, not only as a client, meaning that by default the NTP server is accessible from all hosts in your network. The restrict parameter allows you to limit this access, if needed, and also will limit how the NTP server acts on your server. In the configuration shown in Example 5-27, we have not restricted access in any way, but for further information you can check the ntp.conf man page or:

<http://www.ntp.org/documentation.html>

As soon as the configuration is complete, you can start the NTP daemon in order to start time synchronization (Example 5-28).

---

*Example 5-28 Starting the NTP daemon*

---

```
itsoln1:~ # /etc/init.d/ntp start
Starting network time protocol daemon (NTPD)
done
```

---

Also, set NTPservice to load at boot time, which on SLES or Red Hat Enterprise Linux can be done by using the **chkconfig** utility (Example 5-29).

---

*Example 5-29 Setting up NTP to start at boot time*

---

```
itsoln1:~ # chkconfig ntp on
itsoln1:~ # chkconfig --list ntp
ntp                                0:off 1:off 2:off 3:on  4:off 5:on  6:off
```

---

To check whether the NTP service is running, you can execute the command `/etc/init.d/ntp status`. It returns information about the NTP server (Example 5-30).

*Example 5-30 NTP information provided by /etc/init.d/ntp status*

---

```

itsolnx1:~ # /etc/init.d/ntp status
      remote          refid      st t when poll reach  delay  offset  jitter
=====
*9.56.248.20      9.56.248.88      2 u   76  128  377    0.912    1.230    1.380

```

---

## 5.2.2 NTP and different hardware clocks

By using NTP, servers in the environment are able to keep their clocks synchronized. In Example 5-31, there are two Linux servers, both running NTP clients, on different CPCs. One of these CPCs has a wrong time set (ITSOSS11) and the other is using the current time (ITSOSS12). We executed the `date` and `query time` commands at the same time on both to check the time on each system and compare.

*Example 5-31 Two different Linux servers using NTP on different CPCs*

---

```

itsolnx1:~ # vmcp q userid; vmcp q time; date
ITSOLNX1 AT ITSOS11
TIME IS 17:28:13 EDT MONDAY 06/18/12 #THIS IS z/VM TIME
CONNECT= 02:42:22 VIRTCPU= 002:11.03 TOTCPU= 002:14.40
Mon Jun 18 16:22:34 EDT 2012 #THIS IS LINUX TIME

itsolnx2:~ # vmcp q userid; vmcp q time; date
ITSOLNX2 AT ITSOS12
TIME IS 16:22:34 EDT MONDAY 06/18/12 #THIS IS z/VM TIME
CONNECT= 99:59:59 VIRTCPU= 268:11.03 TOTCPU= 272:41.27
Mon Jun 18 16:22:34 EDT 2012 #THIS IS LINUX TIME

```

---

As you can see, even with ITSOS11 set to a wrong time, ITSOLNX1 could fetch the correct time from NTP, avoiding the data integrity issues on the DB2 side that we had discussed in 5.1.2, “Relocating DB2 when using the hardware clock” on page 41.

Even when you relocate, NTP will always keep your time synchronized. Using NTP and STP is highly recommended in an SSI environment to avoid data integrity issues during relocations.

## 5.3 Final Results

Using the hardware clock:

- ▶ Even with different time zones on each CPC, Linux was able to handle the time difference and adjust its time accordingly. It did not affect DB2 during relocation.
- ▶ Guest relocation does not affect DB2 unless you have different time settings on each CPC *and* execute a logoff/logon on the SSI cluster member.

Although we had data integrity issues, we were able to back up, restore, and rollforward on DB2 V.10.1

Using NTP:

- ▶ We could not find any issues using NTP in our tests. Even with different time configurations on each CPC, Linux was able to synchronize the time, thereby avoiding data integrity issues in DB2.
- ▶ The use of STP and NTP is highly recommended in an SSI cluster environment.





# Networking

In this chapter we test guest relocation in a layer 2 environment, how it affects MAC addresses and how it could affect DB2. In another test, we use SCP to copy a very large file from one Linux on System z guest to another Linux on System z guest and then relocate the guest to other members in the SSI cluster. We will be changing the Maximum Transmission Unit (MTU) size on two guests to 1492 and 8992 (jumbo frame), and then run an SQL query from guest 1 to retrieve data from DB2 on guest 2.

## 6.1 System diagram of a two-system cluster

Our layer 2 test was done on a new two-member SSI cluster, shown in Figure 6-1. We relocated the DB2 v10.1 guests using MAC addresses.

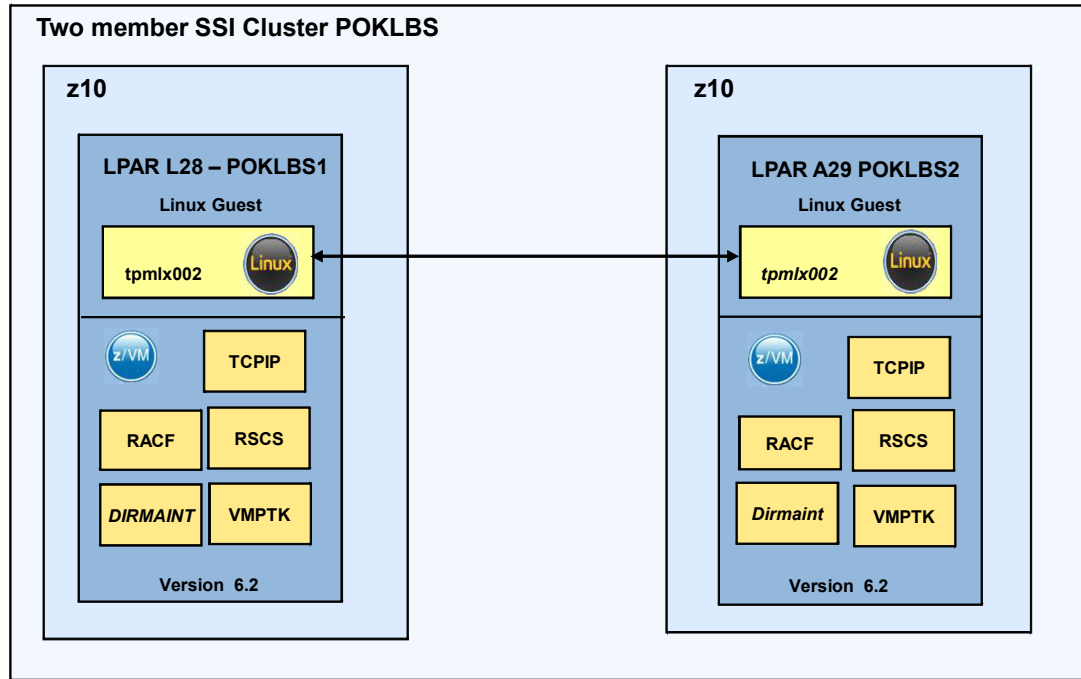


Figure 6-1 Overview of a two-member SSI cluster

## 6.2 Examining how virtual local area networks (VLANs) behave

In order for VSWITCH definitions to be relocatable, there are a few rules that must be followed.

- OSAs must be on the same LAN segment. In Example 6-1, both members are using VLAN 507.

Use QUERY VMLAN to determine the current status of guest LAN activity on the CP system. The response indicates the current value of the system-wide guest LAN attributes, and the number of guest LANs and virtual switches currently defined in the system. Output of this command from z/VM, **query vmlan**, is shown in Example 6-1.

*Example 6-1 QUERY VMLAN*

```
POKLBS1
ETH0  inet addr: 9.12.22.250 mask: 255.255.255.0
      UP MTU: 1500
      vdev: 0140 type: QDIO ETHERNET portname: UNASSIGNED portnumber: 0
      transport type: ETHERNET MAC address: 02-1B-00-00-00-05
      ipv6: DISABLED
      vlan: 507
      cpu: 0 forwarding: ENABLED ipv4 path MTU discovery: DISABLED
      RX bytes: 328021732 TX bytes: 24095533
POKLBS2
```

```

ETH0      inet addr: 9.12.22.251 mask: 255.255.255.0
          UP MTU: 1500
          vdev: 0140 type: QDIO ETHERNET portname: UNASSIGNED portnumber: 0
          transport type: ETHERNET MAC address: 02-1A-00-00-00-05
          ipv6: DISABLED
          vlan: 507
          cpu: 0 forwarding: ENABLED ipv4 path MTU discovery: ENABLED
          RX bytes: 591112707 TX bytes: 4041349

```

- Each z/VM system must have a different MACPREFIX but have the same USERPREFIX. The prefixes are defined through MACPREFIX, USERPREFIX, or both of these options on the VMLAN statement in the system configuration file (Example 6-2).

*Example 6-2 System Configuration file*

---

```

POKLBS1:    VMLAN MACPREFIX 021B00
POKLBS2:    VMLAN MACPREFIX 021A00

```

---

- OSAs must have the same EQID. This can be done through the system configuration file or with the SET command (Example 6-3).

*Example 6-3 System Config*

---

```

RDEVICE 144-146  EQID 123  TYPE OSA
RDEVICE 147-149  EQID 456  TYPE OSA

```

---

If VLAN's are set up correctly on each member, relocating a Linux on System z guest to another member is transparent.

## 6.3 VSWITCH MAC ID

The VMLAN MACPREFIX statement in the system configuration file (SYSTEM CONFIG) defines the first three bytes of the network interface cards (NICs) for each member that is in the SSI cluster. MACPREFIX must be unique for each member. If you have multiple members, increment this value to avoid problems.

The SSI assignment of MAC addresses to network interface cards (NICs) is managed across the SSI cluster. The SSI cluster prevents use of any duplicated MAC addresses. When Linux on System z guests are relocated, the MAC address stays intact without any disruption.

Example 6-4 shows how the network is set up. We then performed a live guest relocation of a Linux on System z guest using MAC addresses or layer 2. We show the system messages produced. We use a two-member SSI cluster in this example. The SYSTEM CONFIG file is located on the PMAINT CF0 mdisk.

*Example 6-4 SYSTEM CONFIG*

---

```

POKLBS1:    VMLAN MACPREFIX 021B00
POKLBS2:    VMLAN MACPREFIX 021A00

```

```

RDEVICE 144-146  EQID 123  TYPE OSA
RDEVICE 147-149  EQID 456  TYPE OSA

```

---

To find out where the Linux on System z guest is running (which member on the SSI cluster), issue the command CP QUERY TPMLX002 at all on z/VM (where tpmlx002 is the Linux on the System z guest on zVM). The output of this command is shown in Example 6-5.

*Example 6-5 Output of query*

---

POKLBS2 : TPMLX002 - DSC

---

Compare the VSWITCHES on both members. They both should be called NET9, defined as ETHERNET, using VLAN 0507, and the OSA address 0144, should have an EQID of 123. The MAC addresses will be different. TPMLX002 is listed as one of the adaptor owners and is the Linux on System z guest we will be relocating. As shown in Example 6-6, the MAC address of TPMLX002 is 02-1A-00-00-00-0A

Issue the following command from z/VM:

CP QUERY VSWITCH NET9 DETAILS

*Example 6-6 Display MAC addresses*

---

VSWITCH SYSTEM NET9 Type: QDIO Connected: 1 Maxconn: INFINITE  
PERSISTENT RESTRICTED **ETHERNET** Accounting: OFF  
USERBASED  
VLAN Aware Default **VLAN: 0507** Default Porttype: Access GVRP: Enabled  
Native VLAN: 0001 VLAN Counters: OFF  
MAC address: **02-1A-00-00-00-02** MAC Protection: Unspecified  
State: Ready  
IPTimeout: 5 QueueStorage: 8  
Isolation Status: OFF  
Uplink Port:  
RDEV: 0144.P00 VDEV: 0603 Controller: DTCVSW2  
**EQID: 123**  
Uplink Port Connection:  
RX Packets: 859151 Discarded: 7409 Errors: 0  
TX Packets: 507700 Discarded: 0 Errors: 0  
RX Bytes: 1098187905 TX Bytes: 143171662  
Device: 0603 Unit: 000 Role: DATA Port: 2049  
Adapter Connections:  
Adapter Owner: **TPMLX002** NIC: 0600.P00 Name: linux Type: QDIO  
Porttype: Access  
RX Packets: 26250 Discarded: 0 Errors: 0  
TX Packets: 276 Discarded: 0 Errors: 0  
RX Bytes: 3227947 TX Bytes: 38789  
Device: 0602 Unit: 002 Role: DATA Port: 0018  
VLAN: 0507  
Options: Ethernet Broadcast  
Unicast MAC Addresses:  
**02-1A-00-00-00-0A**  
Multicast MAC Addresses:  
01-00-5E-00-00-01  
33-33-00-00-00-01  
33-33-00-00-02-02  
33-33-FF-00-00-0A

---



In Example 6-6, VSWITCH NET9 is defined as a layer 2, ETHERNET, VLAN 507, with a MAC address of 02-1A-00-00-00-02. The 00-00-02 was defined by the CP.

Linux on System z guest TPMLX002 has been assigned a MAC address of 02-1A-00-00-00-0A .

What makes this VSWITCH relocatable is the EQID which is set to 123.

This must be set to the same value on all SSI members of the cluster.

To ensure that VSWITCH NET9 is defined properly on POKLBS1:

1. Log on to maint on PSOLBS1.
2. Issue this command from z/VM: CP QUERY VSWITCH NET9 DETAILS (the output is shown in Example 6-7).

*Example 6-7 Display from the z/VM command*

---

```
VSWITCH SYSTEM NET9  Type: QDIO  Connected: 1  Maxconn: INFINITE
PERSISTENT RESTRICTED  ETHERNET          Accounting: OFF
USERBASED
VLAN Aware Default VLAN: 0507  Default Porttype: Access  GVRP: Enabled
      Native VLAN: 0001  VLAN Counters: OFF
MAC address: 02-1A-00-00-00-02  MAC Protection: Unspecified
State: Ready
IPTimeout: 5      QueueStorage: 8
Isolation Status: OFF
Uplink Port:
RDEV: 0144.P00 VDEV: 0603 Controller: DTCVSW2
EQID: 123
Uplink Port Connection:
RX Packets: 859213  Discarded: 7424  Errors: 0
TX Packets: 507755  Discarded: 0      Errors: 0
RX Bytes: 1098195017      TX Bytes: 143182972
Device: 0603 Unit: 000  Role: DATA  Port: 2049
Adapter Connections:
Adapter Owner: TPMLX002 NIC: 0600.P00 Name: linux  Type: QDIO
Porttype: Access
RX Packets: 13      Discarded: 0      Errors: 0
TX Packets: 0      Discarded: 0      Errors: 0
RX Bytes: 1990      TX Bytes: 0
Device: 0602 Unit: 002  Role: DATA  Port: 0020
VLAN: 0507
Options: Ethernet Broadcast
Unicast MAC Addresses:
      02-1A-00-00-00-0A
Multicast MAC Addresses:
      01-00-5E-00-00-01
      33-33-00-00-00-01
      33-33-00-00-02-02
      33-33-FF-00-00-0A
```

---

In the above example, VSWITCH NET9 is defined as a layer 2, ETHERNET, with a MAC address of 02-1B-00-00-00-02. The 00-00-02 was defined by the CP. It also has the same EQID of 123. As far as the network is concerned, TPMLX002 is a guest that is relocatable.

To further verify this:

1. Login to TPMLX002 using SSH
2. Issue the command **ifconfig** from SSH. The output is shown in Example 6-8.

*Example 6-8 Display from ifconfig command*

---

```
eth0      Link encap:Ethernet  HWaddr 02:1A:00:00:00:0A
          inet addr:9.12.22.51  Bcast:9.12.22.255  Mask:255.255.255.0
          inet6 addr: fd55:faaf:elab:20e:1a:ff:fe00:a/64  Scope:Global
          inet6 addr: fe80::1a:ff:fe00:a/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1492  Metric:1
          RX packets:105606 errors:0 dropped:0 overruns:0 frame:0
          TX packets:987 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:11086276 (10.5 Mb)  TX bytes:98460 (96.1 Kb)
```

---

TPMLX002 shows a mac address of 02:1A:00:00:00:0A.

From TPMLX002:

1. Issue the command **tail -f /var/log/messages** from the SSH session.
2. From userid maint on POKLBS2, issue the command **vmrelocate test tpmlx002 poklbs1** from z/VM. The output for this is shown in Example 6-9.

*Example 6-9 Output from the relocation test command*

---

```
User TPMLX002 is eligible for relocation to POKLBS1
```

---

To perform the relocation, issue the following command in z/VM:

```
vmrelocate move tpmlx002 poklbs1
```

The output is shown in Example 6-10.

*Example 6-10 Output from the relocation command*

---

```
Relocation of TPMLX002 from POKLBS2 to POKLBS1 started
User TPMLX002 has been relocated from POKLBS2 to POKLBS1
```

---

From the SSH session opened for TPMLX002, issue the following command:

```
/var/log/messages
```

The output is shown in Example 6-11.

*Example 6-11 Output of /var/log/messages*

---

```
kernel: qeth.aeb579: 0.0.0600: The qeth device driver failed to recover an error
on the device
kernel: qeth: irb 00000000: 00 c2 40 17 7e 88 f0 38 0e 02 00 00 00 80 00 00
..@.~..8.....
kernel: qeth: irb 00000010: 01 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00
.....
kernel: qeth: sense data 00000000: 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
.....
kernel: qeth: sense data 00000010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
.....
```

```

kernel: qeth.aeb579: 0.0.0600: The qeth device driver failed to recover an error
on the device
kernel: qeth: irb 00000000: 00 c2 40 17 7e 88 f0 38 0e 02 00 00 00 80 00 00
..@.~..8.....
kernel: qeth: irb 00000010: 01 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00
.....
kernel: qeth: irb 00000020: 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
.....
kernel: qeth: irb 00000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
.....
kernel: qeth.dfa988: 0.0.0600: A recovery process has been started for the device
kernel: qdio: 0.0.0602 OSA on SC 2 using AI:1 QESM:0 PCI:1 TDD:1 SIGA:RW A
kernel: qeth.980814: 0.0.0600: MAC address 02:1a:00:00:00:0a successfully
registered on device eth0
kernel: qeth.7c6a37: 0.0.0600: Device is a Guest LAN QDIO card (level: V620)
kernel: with link type GuestLAN QDIO (portname: linux)
kernel: qeth.b3618e: 0.0.0600: Device successfully recovered!
skernel: with link type GuestLAN QDIO (portname: linux)
kernel: qeth.b3618e: 0.0.0600: Device successfully recovered!

```

---

The messages seen in Example 4-11 appear each time the Linux on System z guest is relocated. The failure and recovery process is part of the relocation process and is not a problem. The corresponding network interface is kept, but the carrier is switched off during recovery. Usually, connections are just delayed and no packets are lost.

Table 6-1 shows that the MAC address changes only after shutting down and logging off the Linux on System z/VM userid. If we shut down but do not log off the userid from z/VM, and restart the Linux on System z, the MAC address does not change. We have already shown in the previous test that relocating a Linux on System z guest does not change the MAC address.

*Table 6-1 MAC address after shutting down and logging off*

zVM SSI system	Linux on z System	starting MAC address	Action	Ending Mac Address
POKLBS2	tpmlx002	02:1A:00:00:00:0A	Relocated to POKLBS1	02:1A:00:00:00:0A
POKLBS1	tpmlx002	02:1A:00:00:00:0A	Linux shutdown and rebooted. z/VM id not logged off	02:1A:00:00:00:0A Note: did not pick up POKLBS1 PREMAC address.
POKLBS1	tpmlx002	02:1A:00:00:00:0A	Linux shut down and z/VM ID logged off, logged back on and restarted the Linux system.	02:1B:00:00:00:6E
POKLBS1	tmplx002	02:1B:00:00:00:6E	relocated to POKLBS2	02:1B:00:00:00:6E

In these tests we have shown that when DB2 is running on Linux on System z in an SSI cluster, the MAC address does not change when the system is relocated except when the Linux guest on z/VM is logged off. Also, the failed and recovery messages caused by the relocation shown in /var/log/messages were not in error, but part of the relocating process.

## 6.4 Changing to jumbo frames and relocating DB2

OSA-Express2 and later allow setting the Maximum Transmission Unit (MTU) to 8992 (jumbo frames). In order for two guests to benefit from jumbo frames, all networking devices between the guests (inclusive), should have their MTU set to 8992.

Figure 6-2 shows VSWITCH1 on each CPC, having an MTU size of 1492. VSWITCH2 on each CPC is capable of handling jumbo frames.

Interface ETH0 was set up to go through VSWITCH1, and ETH1 was set up to go through VSWITCH2. For the tests that follow, only ETH1 was used; ETH0 was taken down.

The intention of this test was not to generate benchmark information.

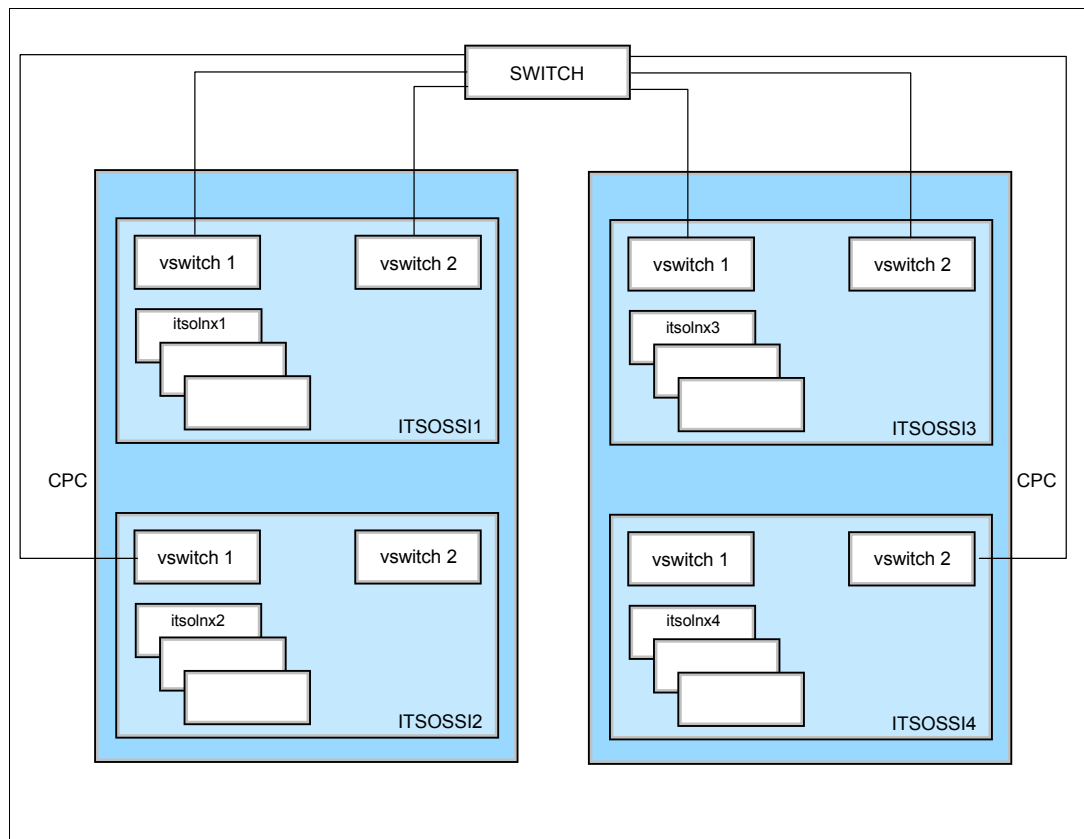


Figure 6-2 Network layout

The following test scenarios show how DB2 behaves during a relocation using MTU sizes of 1492 and 8992. For a simple reference, an SCP (secure copy) transfer test was also executed.

## 6.4.1 Environment

Figure 6-3 illustrates the setup and relocation scheme.

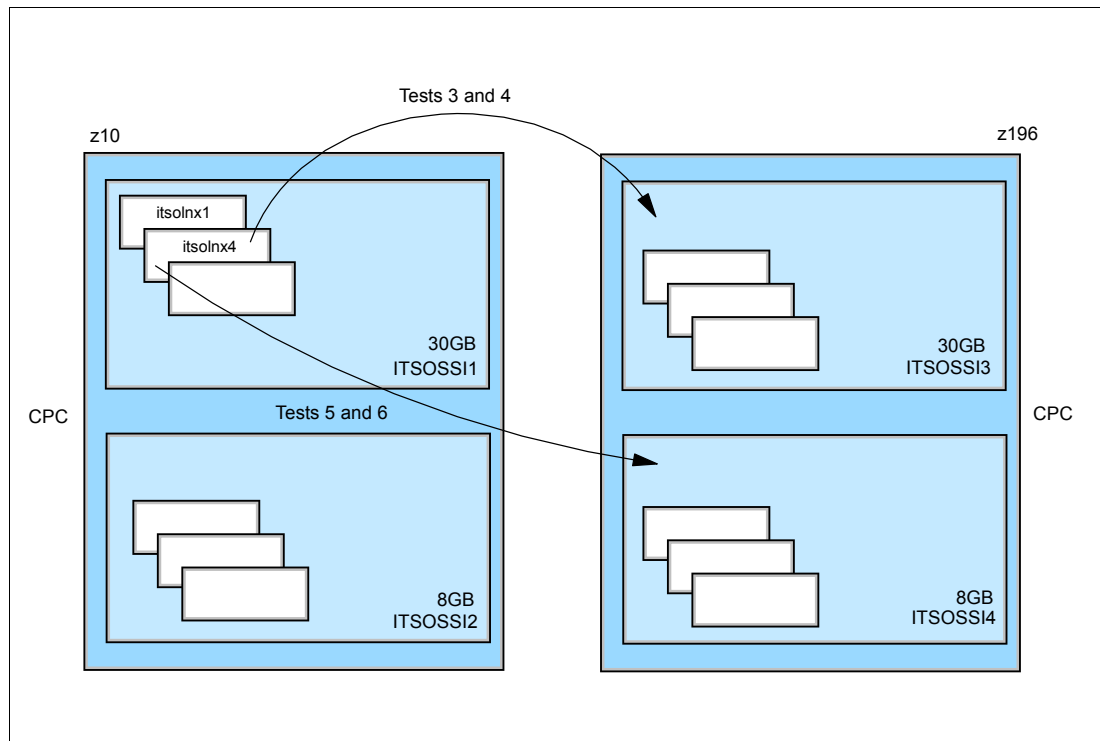


Figure 6-3 Environment, with indication of where itsolnx4 is relocated to

### System environment

- ▶ Member 1 of SSI cluster z/VM - node itsossi1, 30 GB memory
- ▶ Member 3 of SSI cluster z/VM - node itsossi3, 30 GB memory
- ▶ Member 4 of SSI cluster - z/VM - node itsossi4, 8 GB memory
- ▶ Guest 1 - node itsolnx1 (IP address 9.12.5.118), 20 GB virtual memory
- ▶ Guest 2 - node itsolnx4 (IP address 9.12.5.119), 20 GB virtual memory
- ▶ DB2 DB resides on guest 2 - approximately 510,000 records, each record with approximately 3000 bytes
- ▶ SQL queries on guest 1: **select \* from table name**

## 6.4.2 Tests

### Test scenarios running a SELECT query

Linux on System z guest named ITSOLNX1 had its DB2 configured to access the remote database of ITSOLNX4, so that a SELECT query could be run from ITSOLNX1 to retrieve data from ITSOLNX4. In Example 6-12, we enabled access to ITSOLNX4's database from ITSOLNX1.

#### Example 6-12 Enabling access to remote database

```
db2inst1@itsolnx1:~> db2 catalog tcpip node itsolnx4 remote 9.12.5.119 server
60004 remote_instance db2inst1 ostype Linux
db2inst1@itsolnx1:~> db2 catalog database sample as "rsample" at node itsolnx4
db2inst1@itsolnx1:~> db2 connect to rsample USER db2inst1 USING db2inst1
```

The MTU size was set using the IP utility, as shown in Example 6-13.

*Example 6-13 Setting the MTU to 8992 on the Linux guests*

```
# on guest 1
itsolnx1:~ # ip link set dev eth1 mtu 8992

# on guest 2
itsolnx4:~ # ip link set dev eth1 mtu 8992
```

The SQL query shown in Example 6-14 was then run for each test, prefixed with “time”, to check how long the query took to run. The results are presented on Table 6-2.

*Example 6-14 SQL query run during the tests*

```
db2inst1@itsolnx1:~> time db2 “select * from tablename” >/dev/null
```

The following were the tests we performed:

1. Test1 - Both guests reside on member 1. MTU size set to 1492 on both guests. Guest 1 issues the SQL query command.
2. Test2 - Both guests reside on member 1. MTU size set to 8992 on both guests. Guest 1 issues the SQL query command.
3. Test3 - Both guests reside on member 1. MTU size is set to 1492. Guest 2 is relocated to member 3 while guest 1 issues the SQL query command.
4. Test4 - Both guests reside on member 1. MTU size is set to 8992. Guest 2 is relocated to member 3 while guest 1 issues the SQL query command.
5. Test5 - Both guests reside on member 1. MTU size is set to 1492. Guest 2 is relocated to member 4 while guest 1 issues the SQL query command.
6. Test6 - Both guests reside on member 1. MTU size is set to 8992. Guest 2 is relocated to member 4 while guest 1 issues the SQL query command.

*Table 6-2 Test results*

Test #	Guest	From	To	MTU	Time	Relocation	Processor Load
1	itsoss4	itsossi1	itsossi1	8992	0m14.352s	No	2
2	itsoss4	itsossi1	itsossi1	1492	0m13.186s	No	2
3	itsoss4	itsossi1	itsossi3	8992	0m13.092s	Yes	1
4	itsoss4	itsossi1	itsossi3	1492	0m13.713	Yes	1
5	itsoss4	itsossi1	itsossi4	8992	0m13.322	Yes	1
6	itsoss4	itsossi1	itsossi4	1492	0m14.010s	Yes	1

### ***Test scenarios running an SCP transfer***

A file of approximately 900 MB was copied from guest 1 to guest 2 in each test, using the command shown in Example 6-15. The results are presented in Table 6-3 on page 67.

*Example 6-15 secure copy command executed during tests*

```
db2inst1@itsolnx1:~> time scp largefile.img root@9.12.5.119:~/
```

The tests we performed were:

1. Test 1 - Both guests reside on member 1. MTU size set to 1492 on both guests. Guest 1 SCPs a large file to guest 2.
2. Test 2 - Both guests reside on member 1. MTU size set to 8992 on both guests. Guest 1 SCPs a large file to guest 2.
3. Test 3 - Both guests reside on member 1. MTU size set to 1492 on both guests. Guest 1 SCPs a large file to guest 2. During the SCP operation guest 2 is relocated to member 3.
4. Test 4 - Both guests reside on member 1. MTU size set to 8992 on both guests. Guest 1 SCPs a large file to guest 2. During the SCP operation guest 2 is relocated to member 3.
5. Test 5 - Both guests reside on member 1. MTU size set to 1492 on both guests. Guest 1 SCPs a large file to guest 2. During the SCP operation guest 2 is relocated to member 4.
6. Test 6 - Both guests reside on member 1. MTU size set to 8992 on both guests. Guest 1 SCPs a large file to guest 2. During the SCP operation guest 2 is relocated to member 4.

*Table 6-3 Test results*

Test #	Guest	From	To	MTU	Time	Relocation	Processor Load
1	itsoss4	itsossi1	itsossi1	8992	2m7.391s	No	17
2	itsoss4	itsossi1	itsossi1	1492	3m51.989s	No	11
3	itsoss4	itsossi1	itsossi3	8992	3m16.376	Yes	6
4	itsoss4	itsossi1	itsossi3	1492	3m21.770s	Yes	5
5	itsoss4	itsossi1	itsossi4	8992	3m14.893s	Yes	19
6	itsoss4	itsossi1	itsossi4	1492	3m24.102s	Yes	19

In conclusion, the DB2 tests showed little difference between relocation of members using jumbo frames versus 1492. The SCP copy showed some improvement for jumbo over 1492. However, with a large number of connections jumbo frame setups would likely show better performance when compared to 1492 bytes MTU.







## **Best practices for DB2 v10 during live guest relocation**

In this chapter we discuss some of the best practices for deploying IBM DB2 Version 10 with the IBM System z virtualization technology (z/VM 6.2). Primarily we discuss ways to improve resource utilization for optimizing the performance of DB2 v10, by improving processor utilization and sharing system resources using dynamic resource allocation without rebooting.

## 7.1 Processor sharing

In a clustered environment where DB2 guests are being relocated, using uncapped shared virtual processors is a useful technique for dynamically, triangularly, and automatically managing idle processor capacity to accommodate unpredictable workloads.

Usually CP assigns each virtual machine with at least one virtual processor. CP then allocates processor time to a virtual machine based on its SHARE setting, and by default assigns each virtual machine the same share of processor time.

A general guideline is to define as many virtual processors as needed (maximum processor resources required), but do not exceed the number of real processors assigned to this LPAR. Extra virtual processors just add to the overhead and potentially increase the software multiprocessing factors. For example, if the LPAR has four IFLs, then do not allocate five virtual processors to a single Linux guest machine. If a situation occurs where the Linux guest uses 100% of the processors, that will adversely affect the entire LPAR.

However, in an LPAR with four IFLs, you can assign three virtual processors to one Linux guest and two virtual processors to a second Linux guest, as well as another two virtual processors to a third Linux guest. All requests for processor cycles will be managed by z/VM according to the relative priorities of the Linux guests. A processor configuration best practice is to maintain the ratio of four active virtual processors to one logical processor allocated to the LPAR.

But if the guest system operates at greater than 90% utilization for sustained periods, then we can dedicate a processor for the guest. The system operator can then use the DEDICATE command to dedicate virtual processors to real processors.

**Note:** System operators need to dedicate the processor to a virtual machine only if they are sure that a virtual machine can make full use of the processor.

## 7.2 Dynamic processor allocation - share settings

During a DB2 guest relocation, it is suggested to recalculate the z/VM share setting in the target z/VM cluster before the actual guest relocation. Share setting allows you to control the priority that system resources are assigned to a Linux guest. These resources can include processors, real storage, and I/O capability. An absolute or a relative share can be specified.

Special care needs to be taken when virtual guests are with a relative share, where they receive access to system resources that are in proportion with respect to other virtual machines with relative shares.

**Important:** When a virtual machine is relocated, its current share settings may or may not be appropriate on the destination system. Some thought should be given to what this user's share should be on the new system, and the settings changed using the SET SHARE command if necessary.

### 7.2.1 Share setting scenario

As an example, consider that we have two z/VM clusters (ITSOSS1 and ITSOSS2) with two allocated processors. In ITSOSS1 we have two DB2 Linux guests (ITSOLNX1 and

ITSOLNX2) with share values of 600 and 400, respectively. And in the ITSOSI2 cluster we have two guests (ITSOLNX3 and ITSOLNX4) with 700 and 300, respectively.

### Initial share setting for ITSOSI1 and ITSOSI2

If a virtual guest (ITSOLNX11) has a relative share of 600 on z/VM Cluster1 (ITSOSI1), the normalized IFL allocation would be around one-fourth of the access to system resources in ITSOSI1. And the second virtual guest (ITSOLNX2) has a relative share of 400, where ITSOLNX2 would receives a quarter as much access to system resources as ITSOSI1. See Table 7-1.

Table 7-1 Initial share setting

ITSOSI1 Cluster				ITSOSI2 Cluster			
Guest	z/VM Share Relative	Share % for Guest	Normalized resources for Guest	Guest	z/VM Share Relative	Share % for Guest	Normalized resource for Guest
ITSOLNX1	600	60.00	1.2	ITSOLNX3	800	80.00	1.6
ITSOLNX2	400	40.00	0.8	ITSOLNX4	200	20.00	0.4
	1000				1000		

Now relocate ITSOLNX1 to the target z/VM Cluster ITSOSI2. This would alter the overall sharing of resources setting in the guests. As in Table 7-2, it is evident that the access to system resources for the ITSOLNX1 guest after relocation has doubled the percent of resources shared that it had in the previous cluster.

Table 7-2 After relocation of guests - Share setting

ITSOSI1 Cluster				ITSOSI2 Cluster			
Guest	z/VM Share Relative	Share % for Guest	Normalized resources for Guest	Guest	z/VM Share Relative	Share % for Guest	Normalized Resources for Guest
ITSOLNX1 (Relocated)				ITSOLNX3	800	50.00	1.0
ITSOLNX2	400	100.00	2.0	ITSOLNX4	200	33.33	0.7
	400			ITSOLNX1	600	37.50	0.8
					1600		

Always review the target system share settings prior to the relocation of any DB2 servers.

## 7.3 Memory considerations

When defining memory requirements for virtual Linux guests, remember that the Linux kernel will use all the extra available memory allocated to it as a file system cache. In a shared resource environment such as z/VM, this causes the memory resource to be consumed in the LPAR. Such cases could cause a negative impact greater than the positive impact of I/O avoidance, which is especially true in configurations in which data is shared heavily between virtual servers and is mostly read. Therefore, it is important to assign only the memory needed for the running applications when they are at peak load.

### 7.3.1 z/VM paging

During relocation of guests, especially large database servers, it is always advised to monitor the z/VM paging space. z/VM paging is most efficient when it has large contiguous available space on volumes of the same physical geometry that are dedicated to paging. It is also suggested that page volumes be spread across multiple channels/control units, and over multiple ranks within a storage subsystem, in order to avoid channel bottlenecks and maximize throughput. As a general rule, page space on DASD should not be allowed to eclipse 30% utilization, because the available page space can become very fragmented.

### 7.3.2 Linux swapping

Properly sized Linux guests should not swap or should allow minimal swapping under load. However, all production Linux guests should have some swap space configured to handle unexpected memory demands and prevent Linux from hanging or causing a kernel panic. As with the z/VM paging subsystem, a best practice is to define a hierarchy of swap areas that include:

- ▶ One or two small VDISKS
- ▶ One medium minidisk

#### Use VDISK

A VDISK is a virtual DASD pack that resides in the machine's storage instead of on a real disk somewhere. This is exactly like the concept of a RAM disk that you might be familiar with from other operating systems. Because they exist in storage, VDISKS tend to be a lot faster than normal DASD devices and also a lot smaller.

Include swap devices and prioritize them so that the VDISKS are used first, as shown in Example 7-1.

*Example 7-1 Prioritized swap devices*

/dev/dasdv1	swap	swap	priority=20
/dev/dasdw1	swap	swap	priority=10
/dev/dasdm1	swap	swap	priority=0

In the example, dasdv1 and dasdw1 are virtual disks and dasdm1 is the minidisk-based swap device. VM minidisk cache should always be disabled for all swap minidisks of Linux virtual machines.

**Important:** When using VDISK in z/VM v6.2, the user directory statement for definition the needs to be DEFINE VDISK, then the MDISK FB-512 definition. If VDISK is created via MDISK definition, then the guest will not relocate.

## 7.4 Networking consideration

DB2 guests should be connected to a virtual switch (VSWITCH) in a single system image (SSI) facility. This would make the DB2 guests more flexible for relocation across z/VM clusters. A VSWITCH has many advantages over other available options. It can be connected directly to an OSA-Express QDIO adapter and acts as an extension of the physical switch into the zSeries computer. This capability provides connectivity to the external network without requiring a router. Deployment of a VSWITCH reduces the processor utilization and latency

associated with providing external connectivity through a router virtual machine. The z/VM VSWITCH owns the OSA-Express devices and provides a central point of control and management for network traffic between the guest machines and with the external network.

The VSWITCH can be configured with multiple OSA-Express cards to provide dynamic failover capability. It is generally the recommended virtual network topology.

For solutions involving DB2 database guests communicating with inter and intra LPAR communication, HiperSockets need to be considered. If there is substantial network traffic that flows between LPARs or between Linux virtual machines within a z/VM LPAR, then the use of HiperSockets™ can provide for MTU sizes up to 56 K. Large MTU sizes can dramatically improve network performance, depending on workload characteristics—primarily data streaming workloads.

### **7.4.1 Intraensemble data network**

On zEnterprise® there are solutions where the DB2 servers are implemented on a z/VM v6.2 cluster and the application servers are running on zBX blades. Data communications within the DB2 guests and zBX blade servers are handled through another internal network called the intraensemble data network (IEDN), for which you must define a special OSA channel path ID type called OSA-Express for zBX (OSX). The IEDN allows you to set up VLAN connections between the virtual servers running in the ensemble. From z/VM, the virtual servers attach to the IEDN through virtual switches. The virtual servers can also attach to the external network through the OSA.

It is advised to take advantage of the communication through the IEDN using VSWITCH because it is a network within the machine and therefore more secure than the external network communication.



# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

## IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- ▶ *Up and Running with DB2 on Linux*, SG24-6899
- ▶ *Linux on IBM System z: Performance Measurement and Tuning*, SG24-6926
- ▶ *z/VM and Linux Operations for z/OS System Programmers*, SG24-7603

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

[ibm.com/redbooks](http://ibm.com/redbooks)

## Other publications

These publications are also relevant as further information sources:

- ▶ *z/VM v6R2 Systems Operation*, SC24-6233-02

## Online resources

These websites are also relevant as further information sources:

- ▶ DB2 Version 10.1 for Linux, UNIX, and Windows English manuals:  
<http://www-304.ibm.com/support/docview.wss?uid=swg27024478>
- ▶ Kernel parameter requirements (Linux):  
<http://pic.dhe.ibm.com/infocenter/db2luw/v10r1/>
- ▶ Modifying kernel parameters (Linux):  
<http://pic.dhe.ibm.com/infocenter/db2luw/v10r1/>
- ▶ NTP official documentation:  
<http://www.ntp.org/documentation.html>
- ▶ Performance variables:  
<http://pic.dhe.ibm.com/infocenter/db2luw/v10r1/>
- ▶ database\_memory - Database shared memory size configuration parameter:  
<http://pic.dhe.ibm.com/infocenter/db2luw/v10r1/>

- Tuning hints and tips:

<http://www.ibm.com/developerworks/linux/linux390/perf/>

## Help from IBM

IBM Support and downloads

[ibm.com/support](http://ibm.com/support)

IBM Global Services

[ibm.com/services](http://ibm.com/services)



# Index

## B

buffer pools 24

## C

configuring NTP 52

CTC channels 2

## D

db2mtrk 25

db2pd 22

## E

EQID 60

ETHERNET 60

## I

Inter-System Facility for Communications 2

## L

layer 2 57

loading vmcp module 42

## M

MAC 57

MACPREFIX 59

MTU 57

## N

Network 39

Network Time Protocol 39

NTP 39

## O

ORDER BY 30

OSA-Express2 64

## P

package cache 24

## R

Redbooks website 75

Contact us ix

rollforward 46

## S

SCP 57

Self Tuning Memory Manager 21

self\_tuning\_mem 24

Server Time Protocol 40

SHEAPTHRES\_SHR 27

sort memory 24

SORTHEAP 27

SQLDBCON 22

SSI cluster 61

STP 40

STP\_timestamping 40

STP\_tz 40

SYSTEM CONFIG 59

## T

Time Synchronization

Hardware clock 40

time synchronization 39

changing timezone configuration 41

configuring NTP 52

Network Time Protocol 39

NTP 39

Server Time Protocol 40

STP 40

Test 1

changing timezone configuration 41

Test 2

changing the time manually during IPL 43

## U

UPDATE DATABASE CONFIGURATION 24

USERPREFIX 59

## V

VSWITCH 58

## Z

z/VM page allocation 32

z/VM performance tool kit 32





## DB2 10 for Linux on System z Using z/VM v6.2, SSI Clusters and LGR

(0.2"spine)  
0.17"<->0.473"  
90<->249 pages







# DB2 10 for Linux on System z Using z/VM v6.2, Single System Image Clusters and Live Guest Relocation



**Maximize DB2 10  
availability using SSI  
clusters and LGR**

**Discover the benefits  
of z/VM 6.2 LGR**

**Use DB2 STMM and  
z/VM 6.2 LGR**

IBM z/VM 6.2 introduced significant changes to z/VM with a multisystem clustering technology that allows up to four z/VM instances in a single system image (SSI) cluster. This technology is important because it offers you an attractive alternative to vertical growth by adding new z/VM systems. In the past, this capability required duplicate efforts to install, maintain, and manage each system. With SSI, these duplicate efforts are reduced or eliminated.

Support for live guest relocation (LGR) lets you move Linux virtual servers without disrupting your business or incurring loss of service, thus reducing planned outages. The z/VM systems are aware of each other and take advantage of their combined resources. LGR enables you to relocate guests from a system requiring maintenance to a system that will remain active during maintenance.

A major advantage for DB2 v10 clients is that using z/VM 6.2 does not require any changes to existing DB2 structures. This remarkable benefit is due to the fact that DB2 v10 is installed as part of the Linux guest on z/VM and is fully integrated into LGR. This allows you to smoothly move DB2 v10 when you move Linux virtual servers, without interrupting either DB2 v10 or z/VM operations and services.

This IBM Redbooks publication will help you understand how DB2 10 on Linux for System z behaves while running on a z/VM that is being relocated using z/VM's 6.2 Live Guest Relocation feature.

In this book we explore memory management, the DB2 self-tuning memory manager feature, time synchronization, networking, and storage and performance considerations with regard to relocation. We also offer some best practices found during a live guest relocation for DB2 v10.

## INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

### BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:**  
[ibm.com/redbooks](http://ibm.com/redbooks)