z/OS

System z Platform Test Report for z/OS and Linux Virtual Servers

Version 1 Release 9

SA22-7997-06

z/OS

System z Platform Test Report for z/OS and Linux Virtual Servers

Version 1 Release 9

SA22-7997-06

Note!
Before using this information and the products it supports, be sure to read the general information under "Notices" on page 189.

Seventh Edition, December 2007

This is a major revision of SA22-7997-05.

This edition applies to Parallel Sysplex environment function that includes data sharing and parallelism. Parallel Sysplex uses the z/OS (5694-A01) operating system.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

IBM welcomes your comments. A form for readers' comments may be provided at the back of this publication, or you may address your comments to the following address:

IBM Corporation
Department B6ZH, Mail Station P350
2455 South Road
Poughkeepsie, NY 12601-5400
United States of America

FAX (United States & Canada): 1+845+432-9414
FAX (Other Countries): Your International Access Code +1+845+432-9414

IBMLink™ (United States customers only): IBMUSM(LBCRUZ)
Internet e-mail: lbcruz@us.ibm.com
World Wide Web: www.ibm.com/servers/eserver/zseries/zos/integtst/

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Opening remarks

A message from our team

We changed our title from *z/OS Parallel Sysplex Test Report* but it's still us! Same team, same testing, but we've gradually expanded our focus from Parallel Sysplex® to a platform-wide view of z/OS® and Linux® on System z™ in the enterprise. To reflect that focus, we changed our title to ***IBM® System z Platform Test Report for z/OS and Linux Virtual Servers***.

As you read this document, keep in mind that we need your feedback. We want to hear anything you want to tell us, whether it's positive or less than positive. We especially want to know what you'd like to see in future editions. That helps us prioritize what we do in our next test phase. We will also make additional information available upon request if you see something that sparks your interest. To find out how to communicate with us, see "How to send your comments" on page xviii.

We are a team whose combined computing experience is hundreds of years, but we have a great deal to learn from you, our customers. We will try to put your input to the best possible use. Thank you.

Al Alexsa
Loraine Arnold
Ozan Baran
Ryan Bartoe
Duane Beyer
Jeff Bixler
Muriel Bixler
Dave Buehl
Jon Burke
Alex Caraballo
Phil Chan
John Corry
Don Costello
Kevin Coyne
Luis Cruz
Lisa Dodaro
Eli Dow
Bob Fantom
Nancy Finn
Bobby Gardinor
Kieron Hinds
Fred Lates
Al Lease
Frank LeFevre
Dolores Lovalo
Scott Loveland
George Markos
Sue Marcotte
Tammy McAllister
Arvind Mistry
Azeem Mohammed
Elaine Murphy
Al Nims
Jim Rossi
Tom Sirc
Karen Smolar
Anthony Sofia
Jeff Stokes
Jim Stutzman
Lissette Toledo
Ashwin Venkatraman

Important—Currency of the softcopy edition

Each release of the *z/OS Collection* (SK3T-4269 or SK3T-4270) and *z/OS DVD Collection* (SK3T-4271) contains a back-level edition of this test report.

Because we produce our test reports twice a year, June and December, we cannot meet the production deadline for the softcopy collections that coincide with the product's GA release and the softcopy collection refresh date six months later. Therefore, there is normally a one-edition lag between the release of our latest test report edition and the softcopy collection in which it is included. That is, the test report that appears in any given softcopy collection is normally one edition behind the most current edition available on the Web.

If you obtained this document from a softcopy collection on CD-ROM or DVD, you can get the most current edition from the IBM System z Platform Test Report Web site at www.ibm.com/servers/eserver/zseries/zos/integtst/.

Contents

Figures

Tables

About this document

This document is a test report written from the perspective of a system programmer. The IBM System z Platform Evaluation Test (zPET) team (formerly known as the z/OS Integration Test team)—a team of IBM testers and system programmers simulating a customer production Parallel Sysplex environment—wants to continuously communicate directly with you, the mainframe system programmer. We provide this test report to keep you abreast of our efforts and experiences in performing the final verification of each system release before it becomes generally available to customers.

An overview of System z Platform Evaluation Test (zPET)

We have been producing this test report since March, 1995. At that time, our sole focus of our testing was the S/390® MVS™ Parallel Sysplex. With the introduction of OS/390® in 1996, we expanded our scope to encompass various other elements and features, many of which are not necessarily sysplex-oriented. In 2001, OS/390 evolved into z/OS, yet our mission remains the same to this day. In 2005, we expanded to add a Linux Virtual Server arm to our overall environment, which will be used to emulate leading-edge customer environments, workloads, and activities.

Our mission and objectives

IBM's testing of its products is and always has been extensive. ***The test process described in this document is not a replacement for other test efforts.*** Rather, it is an additional test effort with a shift in emphasis, focusing more on the customer experience, cross-product dependencies, and high availability. We simulate the workload volume and variety, transaction rates, and lock contention rates that exist in a typical customer shop, stressing many of the same areas of the system that customers stress. When we encounter a problem, our goal is to keep systems up and running so that end users can still process work.

Even though our focus has expanded over the years, our objectives in writing this test report remain as they were:

- Run a Parallel Sysplex in a production shop in the same manner that customers do. We believe that only by being customers ourselves can we understand what our own customers actually experience when they use our products.
- Describe the cross-product and integrated testing that we do to verify that certain functions in specific releases of IBM mainframe server products work together.
- Share our experiences. In short, if any of our experiences turn out to be painful, we tell you how to avoid that pain.
- Provide you with specific recommendations that are tested and verified.

We continue to acknowledge the challenges that information technology professionals face in running multiple hardware and software products and making them work together. We're taking more of that challenge upon ourselves, ultimately to attempt to shield you from as much complexity as possible. The results of our testing should ultimately provide the following benefits:

- A more stable system for you at known, tested, and reproducible service levels

- A reduction in the time and cost of your migration to new product releases and functions.

Our test environment

The Parallel Sysplex that forms the core of our test environment has grown and changed over the years. Today, our test environment has evolved to a highly interconnected, multi-platform on demand enterprise—just like yours.

To see what our environment looks like, see the following:

- "Our Parallel Sysplex hardware configuration" on page 3
- "Our Parallel Sysplex software configuration" on page 10
- "Our workloads" on page 14
- "Our networking configuration" on page 23

Who should read this information

System programmers can use this information to learn more about the integration testing that IBM performs on z/OS and certain related products, including selected test scenarios and their results. We assume that the reader has a working knowledge of MVS and Parallel Sysplex concepts and terminology, and at least a basic level of experience with installing and managing the z/OS operating system, subsystems, network products, and other related software. See "Where to find more information" on page xvii.

How to use this information

Use this test report as a companion to—***never*** *instead of*—your reading of other z/OS element-, feature-, or product-specific documentation. Our configuration information and test scenarios should provide you with concrete, real-life examples that help you understand the "big picture" of the Parallel Sysplex environment. You might also find helpful tips or recommendations that you can apply or adapt to your own situation. Reading about our test experiences should help you to confidently move forward and exploit the key functions you need to get the most from your technology investment.

However, you also need to understand that, while the procedures we describe for testing various tasks (such as installation, configuration, operation, and so on) are based on the procedures that are published in the official IBM product documentation, they also reflect our own specific operational and environmental factors and are intended for illustrative purposes only. Therefore, ***do not*** use this document as your sole guide to performing any task on your system. Instead, follow the appropriate IBM product documentation that applies to your particular task.

How to find our test reports

We make all editions of our test reports available on our z/OS Integration Test Web site at:

www.ibm.com/servers/eserver/zseries/zos/integtst/

If you cannot get to our Web site for some reason, see "Appendix B. Availability of our test reports" on page 185 for other ways to access our test reports.

We publish our test reports twice a year, every June and December. Our December edition covers our initial test experiences with a new z/OS release, including migration. Our June edition is the final edition for that release; it is cumulative, building upon the December edition with any new test experiences we've encountered since then. We freeze the June edition and begin anew with the next release in December. The most recent edition of our test report, as well as the final editions for previous releases of z/OS, are available on our Web site.

We also have a companion publication, *z/OS V1R8.0 System z Parallel Sysplex Recovery*, GA22-7286. In this publication, we focus on describing:

- How to be prepared for potential problems in a Parallel Sysplex
- What the indicators are to let you know there is a problem
- What actions to take to recover

The recovery scenarios we describe are based on our own experiences in our particular test environment while running z/OS V1R8, DB2® V8, IMS™ V9, WebSphere® Application Server V6.0, WebSphere MQ V6 and CICS® TS V3R1, These scenarios do not represent a comprehensive list of all possible approaches and outcomes, but do represent the approaches we have tested and that work for us.

Note: The recovery book was written in the z/OS V1R8 time frame; however, many of the recovery concepts that we discuss still apply to later releases of z/OS.

Where to find more information

If you are unfamiliar with Parallel Sysplex terminology and concepts, you should start by reviewing the following publications:

Table 1. Parallel Sysplex planning library publications

Publication title	Order number
z/OS Parallel Sysplex Overview	SA22-7661
z/OS MVS Setting Up a Sysplex	SA22-7625
z/OS Parallel Sysplex Application Migration	SA22-7662
z/OS Planning for Installation	GA22-7504

In addition, you can find lots of valuable information on the World Wide Web.

- See the Parallel Sysplex for OS/390 and z/OS Web site at: www.ibm.com/servers/eserver/zseries/pso/
- See the Parallel Sysplex Customization Wizard at: www.ibm.com/servers/eserver/zseries/pso/tools.html
- See the z/OS Managed System Infrastructure (msys) for Operations Web site at: www.ibm.com/servers/eserver/zseries/msys/msysops/
- See the IBM Education Assistant which integrates narrated presentations, Show Me Demonstrations, tutorials, and resource links to help you successfully use the IBM software products at:publib.boulder.ibm.com/infocenter/ieduasst/stgv1r0/index.jsp

How to send your comments

Your feedback is important to us. If you have any comments about this document or any other aspect of Integration Test, you can send your comments by e-mail to:

- lbcruz@us.ibm.com, for questions about z/OS and Parallel Sysplex
- lefevre@us.ibm.com, for questions about Linux on System z

Or, you can use the contact form on our Web site at:

www.ibm.com/servers/eserver/zseries/zos/integtst/

If you are reading the PDF version of this document, you can also submit the Readers' Comments form located at the end of the document.

Be sure to include the document number and, if applicable, the specific location of the information you are commenting on (for example, a specific topic heading or page number).

Part 1. System z Platform Evaluation Test

System z Platform Evaluation Test (zPET) focuses on the z/OS and Parallel Sysplex aspects of our computing environment.

We address such topics as:

- Our hardware, software, networking, and security environments
- Migration to the latest release of the z/OS operating system
- Experiences with new functionality offered in the latest z/OS release
- Experiences with various z/OS data management and transaction management products that exploit Parallel Sysplex and data sharing
- Experiences with various z/OS middleware and application enablement products

Chapter 1. About our Parallel Sysplex environment

In this chapter we describe our Parallel Sysplex computing environment, including information about our hardware and software configurations and descriptions of the workloads we run.

Note: Throughout this document, when you see the term ***sysplex***, understand it to mean a sysplex with a coupling facility, which is a ***Parallel Sysplex.***

Overview of our Parallel Sysplex environment

We run two Parallel Sysplexes, one with 9 members and the other with 3 members that consist of the following:

- Four central processor complexes (CPCs) running z/OS in 12 logical partitions (LPs).

  The CPCs consist of the following machine types:
  - One IBM @server zSeries 900 (z900)
  - One IBM @server zSeries 990 (z990)
  - One IBM System z9 Business Class (z9 BC)
  - One IBM System z9 Enterprise Class (z9 EC)

  The z/OS images consist of the following:
  - Eight production z/OS systems
  - Three test z/OS systems
  - One z/OS system to run TPNS (Our December 1998 test report explains why we run TPNS on a non-production system.)
- Five coupling facilities:
  - One failure-independent coupling facility that runs in a LP on a standalone CPC
  - Four non-failure-independent coupling facilities that run in LPs on three of the CPCs that host other z/OS images in the sysplex
- Two Sysplex Timer® external time references (ETRs)
- Other I/O devices, including ESCON- and FICON-attached DASD and tape drives.

"Our Parallel Sysplex hardware configuration" describes all of the above in more detail.

Outside of the Parallel Sysplex itself, we also have ten LPs in which we run the following:
- Two native Linux images
- Eight z/VM® images that host multiple Linux guest images running in virtual machines

Our Parallel Sysplex hardware configuration

This section provides an overview of our Parallel Sysplex hardware configuration as well as other details about the hardware components in our operating environment.

Overview of our hardware configuration

Figure 1 on page 5 is a high-level, conceptual view of our Parallel Sysplex hardware configuration. In the figure, broad arrows indicate general connectivity between processors, coupling facilities, Sysplex Timers, and other I/O devices; they do not depict actual point-to-point connections.

Just recently we removed the IBM System z9 BC Model 2096-S07 from our plex. This processor was running the z/OS.e V1R8 operating system. The last release to support z/OS.e is V1R8. Given this, and the fact that we will soon be migrating our entire sysplex over to z/OS V1R9, we decided not to continue to run z/OS.e in our environment (for the limited time we were still on V1R8).

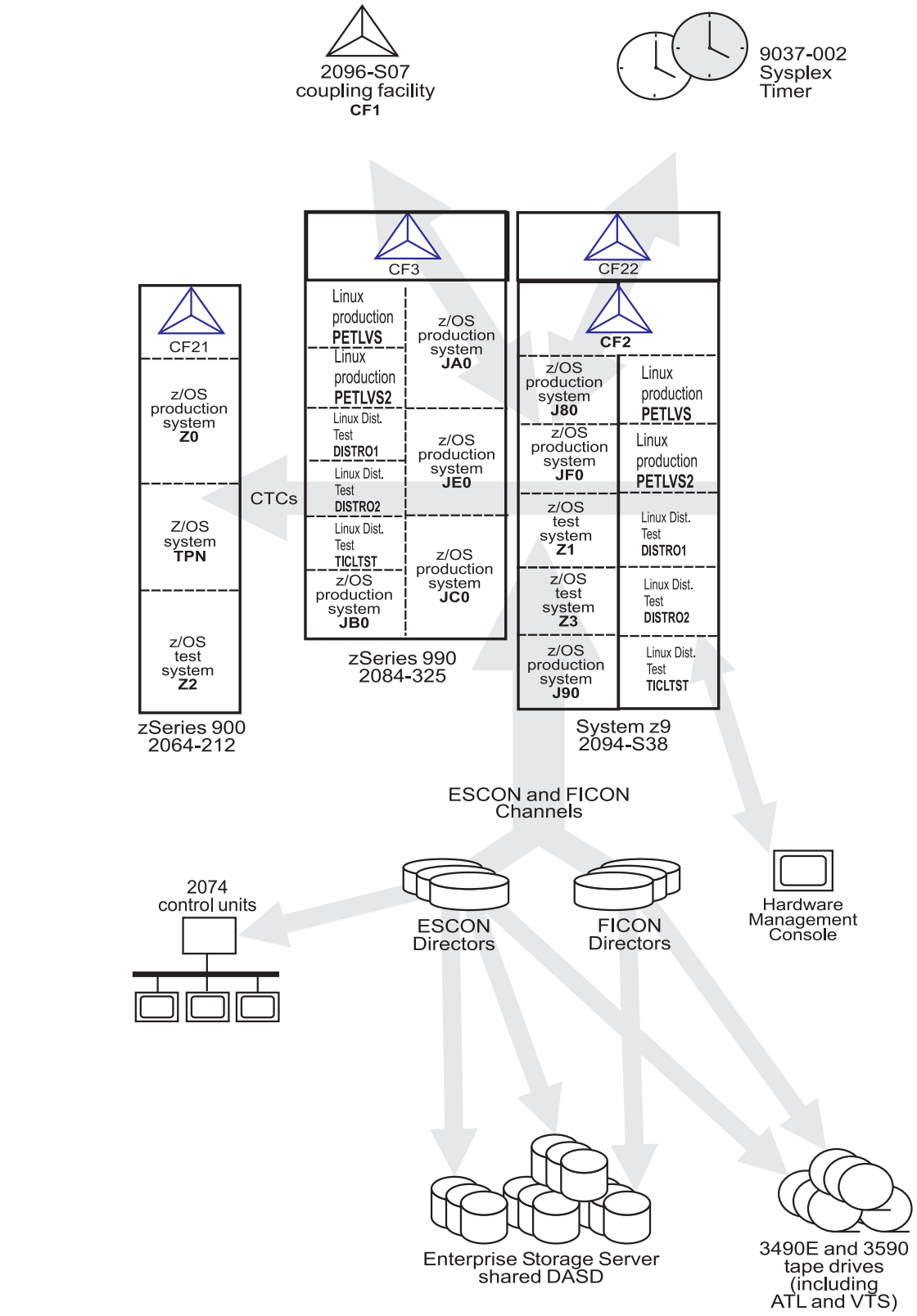


Figure 1. Our sysplex hardware configuration

Hardware configuration details

The figures and tables in this section provide additional details about the mainframe servers, coupling facilities, and other sysplex hardware shown in Figure 1 on page 5.

Mainframe server details

Table 2 provides information about the mainframe servers in our sysplex environment.

Table 2. Our mainframe servers

Server model (Machine type-model)	CPCs CPs	Mode LPs	HSA	Storage	LCSS	System name, usage Sysplex membership CPs, zIIPs, zAAPs Initial LPAR weight
IBM System z9 EC (2094-S38)	1 CPC 38 CPs	LPAR 9 LPs	2176M	112640M	0	**J80**, z/OS production system Plex 1 32 shared CPs 2 shared zIIPs 2 shared zAAPs
				155648M		**J90**, z/OS production system Plex 1 32 shared CPs 2 shared zIIPs 2 shared zAAPs
				15360M	0	**JF0**, z/OS production system Plex 1 16 shared CPs 2 shared zIIPs 2 shared zAAPs
				24576M	0	**Z1**, z/OS test system Plex 2 8 shared CPs 2 shared zIIPs 2 shared zAAPs
				24576M	0	**Z3**, z/OS test system Plex 2 8 shared CPs 2 shared zIIPs 2 shared zAAPs
				3072M	1	**PETLVS**, Linux production system shared CPs weight of 10
				4096M	1	**PETLVS2**, Linux production system 4 shared CPs weight of 10
				3072M	1	**DISTR01**, Linux distribution test system 2 shared IFLs weight of 10
				2048M	1	**DISTR02**, Linux distribution test system 2 shared IFLs weight of 10
				1024M	1	**TICLTST**, Linux distribution test 1 shared IFL weight of 10
IBM @server zSeries 900 Model 212 (2064-212)	1 CPC 16 CPs 4 ICF	LPAR mode 4 LPs (1 LP is a coupling facility)	256M	9216M		**Z0**, z/OS production system Plex 1 8 shared CPs
				6144M		**TPN**, z/OS system for TPNS Plex 1 12 shared CPs
				10752M	0	**Z2**, z/OS test system Plex 2 8 shared CPs

Table 2. Our mainframe servers (continued)

Server model (Machine type-model)	CPCs CPs	Mode LPs	HSA	Storage	LCSS	System name, usage Sysplex membership CPs, zIIPs, zAAPs Initial LPAR weight
IBM @server zSeries 990 Model 325 (2084-325)	1 CPC 32 CPs 2 IFL, 2 zAAP	LPAR mode 20 LPs	1536MB	30720M	2	**JA0**, z/OS production system Plex 1 16 shared CPs, 2 shared zAAPs
				30720M	0	**JB0**, z/OS production system Plex 1 16 shared CPs, 2 shared zAAPs
				22528M	0	**JC0**, z/OS production system Plex 1 16 shared CPs, 2 shared zAAPs
				22528M	2	**JE0**, z/OS production system Plex 1 16 shared CPs, 2 shared zAAPs
				16384M	1	**PETLVS**, Linux production system 4 shared CPs weight of 10
				4096M	1	**PETLVS2**, Linux production system 4 shared CPs weight of 10
				3072M	1	**DISTR01**, Linux distribution test system 2 shared IFLs weight of 10
				2048M	1	**DISTR02**, Linux distribution test system 2 shared IFLs weight of 10
				1024M	1	**TICLTST**, Linux distribution test 1 shared IFL weight of 10

Coupling facility details

Table 3 provides information about the coupling facilities in our sysplex. Figure 1 on page 5 further illustrates the coupling facility channel distribution as described in Table 3.

Table 3. Our coupling facilities

Coupling facility name	Model description CPCs and CPs CFLEVEL (CFCC level) Controlled by	Storage
CF1 (Plex 1)	IBM System Z9 BC Model 2096-S07 stand-alone coupling facility 1 CPC with 4 CPs CFLEVEL=15 (CFCC Release 14.00, Service Level 04.05) Controlled by the HMC	14G
CF2 (Plex 1)	Coupling facility LP on a System z9 (2094-S38) 3 dedicated ICF CPs CFLEVEL=15 (CFCC Release 14.00, Service Level 00.17) Controlled by the HMC	14G
CF3 (Plex 1)	Coupling facility LP on a zSeries 990 Model 325 (2084-325) 3 dedicated ICF CPs CFLEVEL=14 (CFCC Release 14.00, Service Level 00.28) Controlled by the HMC	14G
CF21 (Plex 2)	Coupling facility LP on a zSeries 900 Model 212 (2064-212) 1 dedicated ICF CP CFLEVEL=13 (CFCC Release 13.00, Service Level 04.12) Controlled by the HMC	6G
CF22 (Plex 2)	Coupling facility LP on a System z9 (2094-S38) 1 dedicated ICF CP CFLEVEL=15 (CFCC Release 14.00, Service Level 04.05) Controlled by the HMC	6G

Table 4 illustrates our coupling facility channel configuration on Plex 1.

Table 4. Coupling facility channel configuration on Plex 1

	Coupling facility (CF) images		
Machine type z/OS and CF images	**2096-S07 CF1**	**2094-S38 CF2**	**2084-325 CF3**
2084-325 JA0, JE0, JC0, JB0, CF3	1 CBP 3 CFP	1 CBP 3 CFP	4 CBP 4 CFP
2064-212 Z0, TPN	6 CFP	2 CBP *	4 ICP
2094-S38 J80, JF0, CF2, J90	4 CFP	4 ICP	2 CBP *
* = Same links			

Table 5 illustrates our coupling facility channel configuration on Plex 2.

Table 5. Coupling facility channel configuration on Plex 2

	Coupling facility (CF) Images	
Machine type z/OS and CF images	**2094-S38 CF22**	**2064-212 CF21**
2064-212 Z2, CF21	2 CFP * 1 CBP *	2 ICP
2094-S38 Z1, Z3, CF22	2 ICP	2 CFP * 1 CBP *
* = Same links		

In addition to our coupling facility channel configuration listed in Table 4 on page 8 and Table 5 on page 8, we configured 1 ISC-3 link and 1 ICB-3 link between our 2084-325 CPC and our 2096-S07 CPC which will be used as Server Time Protocol (STP) timing-only links in our new STP environment. See "Migrating to a Server Time Protocol Coordinated Timing Network (CTN)" on page 10 for a further description of STP timing-only links.

Other sysplex hardware details

Table 6 highlights information about the other hardware components in our sysplex.

Table 6. Other sysplex hardware configuration details

Hardware element	Model or type	Additional information
External Time Reference (ETR)	Sysplex Timer (9037-002 with feature code 4048)	We use the Sysplex Timer with the Expanded Availability feature, which provides two 9037 control units connected with fiber optic links. We don't have any Sysplex Timer logical offsets defined for any of the LPs in our sysplex.
Channel subsystem	CTC communications connections	We have CTC connections from each system to every other system. We now use both FICON® and ESCON® CTC channels on all of our CPCs. **Note:** All of our z/OS images use both CTCs and coupling facility structures to communicate. This is strictly optional. You might choose to run with structures only, for ease of systems management. We use both structures and CTCs because it allows us to test more code paths. Under some circumstances, XCF signalling using CTCs is faster than using structures. See *S/390 Parallel Sysplex Performance* for a comparison.
	Coupling facility channels	We use a combination of ISC, ICB, and IC coupling facility channels in peer mode. We use MIF to logically share coupling facility channels among the logical partitions on a CPC. We define at least two paths from every system image to each coupling facility, and from every coupling facility to each of the other coupling facilities.
	ESCON channels	We use ESCON channels and ESCON Directors for our I/O connectivity. Our connections are "any-to-any", which means every system can get to every device, including tape. (We do not use any parallel channels.)
	FICON channels	We have FICON native (FC) mode channels from all of our CPCs to our Enterprise Storage Servers and our 3590 tape drives through native FICON switches. (See *FICON Native Implementation and Reference Guide*, SG24-6266, for information about how to set up this and other native FICON configurations.) We maintain both ESCON and FICON paths to the Enterprise Storage Servers and 3590 tape drives for testing flexibility and backup. Note that FICON channels do not currently support dynamic channel path management. We have also implemented FICON CTCs, as described in the IBM Redpaper *FICON CTC Implementation* available on the IBM Redbooks™ Web site.
DASD	Enterprise Storage Server(R) (ESS, 2105-F20, 800) IBM System Storage (DS6000, DS8000)	All volumes shared by all systems; about 90% of our data is SMS-managed. We currently have four IBM TotalStorage® Enterprise Storage Servers, of which two are FICON only, and two that are attached with both ESCON and FICON. **Note:** Do not run with both ESCON and FICON channel paths from the same CPC to a control unit. We have some CPCs that are ESCON-connected and some that are FICON-connected.
Tape	3490E tape drives	16 IBM 3490 Magnetic Tape Subsystem Enhanced Capability (3490E) tape drives that can be connected to any system.
	3590 tape drives	4 IBM TotalStorage Enterprise Tape System 3590 tape drives that can be connected to any system.

Table 6. Other sysplex hardware configuration details (continued)

Hardware element	Model or type	Additional information
Automated tape library (ATL)	3494 Model L10 with 16 Escon and Ficon attached 3590 tape drives and 8 3592 (Encryption capable) tape drives	All tape drives are accessible from all systems.
Virtual Tape Server (VTS)	3494 Model L10 with 32 virtual 3490E tape drives.	All tape drives are accessible from all systems.

Migrating to a Server Time Protocol Coordinated Timing Network (CTN)

We migrated to the Server Time Protocol (STP) since our last report. Details on this migration can be found on Our latest tips and experiences website. For more information, see the section "Migrating to a Server Time Protocol Configuration" at:

`www.ibm.com/servers/eserver/zseries/zos/integtst/tips.html`

where we discuss the z/OS Integration Test team's experiences associated with that migration.

We begin by first presenting a brief overview of STP and the respective terminology, and then follow it with a high level overview of the z/OS Integration Test lab environment.

We also present both the planning considerations and the actual migration steps taken by the team to deploy STP in their data center.

Our Parallel Sysplex software configuration

We run the z/OS operating system along with the following software products:

- CICS Transaction Server (CICS TS) V3R1
- IMS V9 (and its associated IRLM)
- DB2 UDB for z/OS and OS/390 V8 (and its associated IRLM)
- DB2 UDB for z/OS and OS/390 V9.1 (and its associated IRLM)
- WebSphere for z/OS V6.0.2
- WebSphere MQ for z/OS V6
- Websphere Message Broker V6

Note that we currently only run IBM software in our sysplex.

A word about dynamic enablement: As you will see when you read *z/OS Planning for Installation*, z/OS is made up of base elements and optional features. Certain elements and features of z/OS support something called *dynamic enablement*. When placing your order, if you indicate you want to use one or more of these, IBM ships you a tailored IFAPRDxx parmlib member with those elements or features enabled. See *z/OS Planning for Installation* and *z/OS MVS Product Management* for more information about dynamic enablement.

Overview of our software configuration

Figure 2 on page 11 shows a high-level view of our sysplex software configuration.

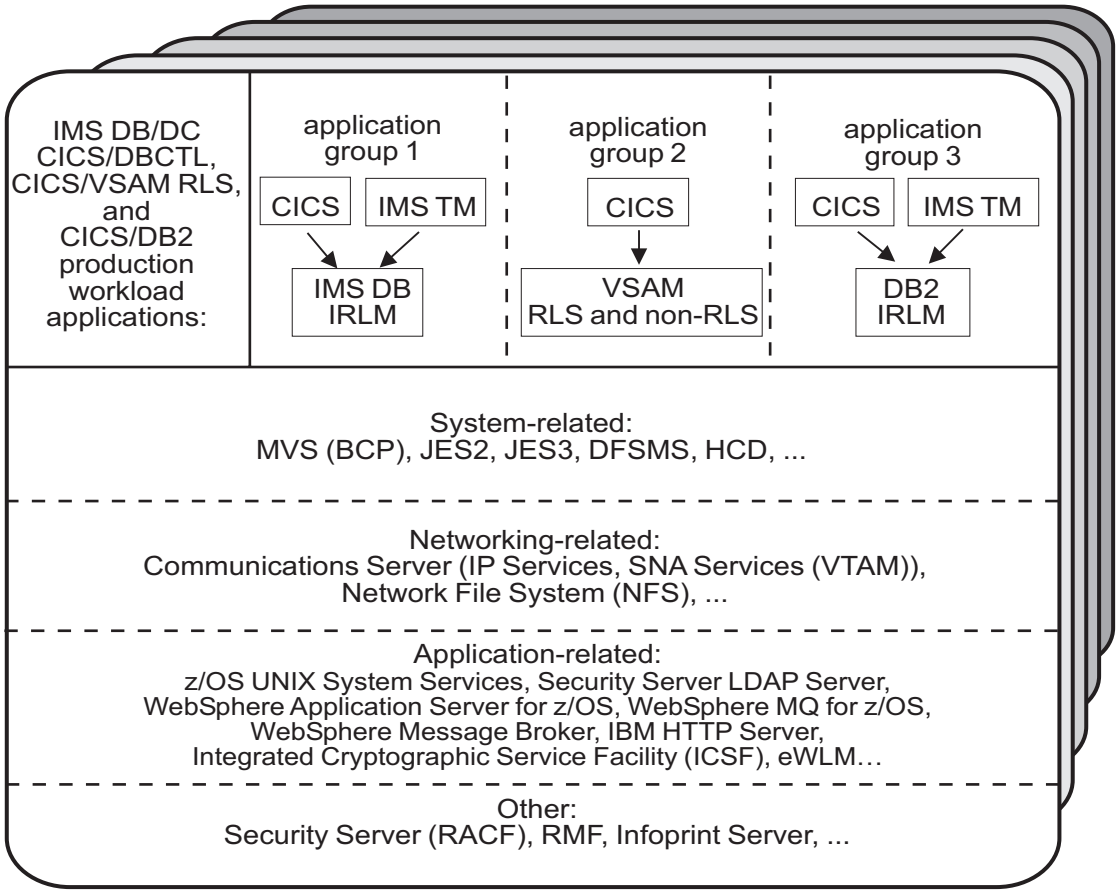


Figure 2. Our sysplex software configuration

We run three separate application groups in one sysplex and each application group spans multiple systems in the sysplex. Table 7 provides an overview of the types of transaction management, data management, and serialization management that each application group uses.

Table 7. Our production OLTP application groups

Application groups	Transaction management	Data management	Serialization management
Group 1	CICS IMS TM	IMS DB	IRLM
Group 2	CICS	VSAM	VSAM record-level sharing (RLS)
Group 3	CICS IMS TM	DB2	IRLM

Our December 1995 test report describes in detail how a transaction is processed in the sysplex using application group 3 as an example. In the example, the transaction writes to both IMS and DB2 databases and is still valid for illustrative purposes, even though our application group 3 is no longer set up that way. For more information about the workloads that we currently run in each of our application groups, see "Database product OLTP workloads" on page 20.

About our naming conventions

We designed the naming convention for our CICS regions so that the names relate to the application groups and system names that the regions belong to. This is important because:

- Relating a CICS region name to its application groups means we can use wildcards to retrieve information about, or perform other tasks in relation to, a particular application group.
- Relating CICS region names to their respective z/OS system names means that subsystem job names also relate to the system names, which makes operations easier. This also makes using automatic restart management easier for us — we can direct where we want a restart to occur, and we know how to recover when the failed system is back online.

Our CICS regions have names of the form CICS*grsi* where:

- *g* represents the application group, and can be either 1, 2, or 3
- *r* represents the CICS region type, and can be either A for AORs, F for FORs, T for TORs, or W for WORs (Web server regions)
- *s* represents the system name, and can be 0 for system Z0, 8 for J80, 9 for J90, and A for JA0 through G for JG0
- *i* represents the instance of the region and can be A, B, or C (we have 3 AORs in each application group on each system)

For example, the CICS region named CICS2A0A would be the first group 2 AOR on system Z0.

Our IMS subsystem jobnames also correspond to their z/OS system name. They take the form IMS*s* where *s* represents the system name, as explained above for the CICS regions.

Using TPC-R V3.3 in our zPET environment

We have added the IBM TotalStorage Productivity Center for Replication (TPC-R) for System z to our lineup of products in our z/OS integration test environment.

In addition to our testing, we have also exploited TPC-R for our own use for:

- Configuring, monitoring and controlling our advanced storage replication services (DS8000s)
- Peer-to-Peer Remote Copy (PPRC)
- FlashCopy

Overall, TPC-R simplified our configuration and we have been very happy with the product. Once up and running, we found the Web browser interface to be intuitive and very easy to use. TPC-R improved our operations support by providing different features such as overwrite protection and improved monitoring and messages. It provided simple operational control of copy services tasks, which includes starting, suspending, and resuming sessions. Our only real trouble area was in the installation and setup (see "Installing and setting up TPC-R" on page 13 for more about this).

TPC-R environment in zPET

We used the following software and hardware during our testing:

- z/OS V1.8 and V1R9
- TPC-R V3.3
- DB2 Version 9.1 (sharing between 3 systems)

- Websphere Application Server V6.1
- DFSMSsdm w/APAR OA18953
- IBM System Storage DS8000 (2107), microcode level 2.4 or higher
- IBM TotalStorage Enterprise Storage Server (ESS) Model 800 (2105-800), LIC level 2.4.4.45 or higher

All the DB2 application and database volumes that were designated for copy to a secondary site were part of a primary IBM System Storage DS8000 and capable of copy services functions. We also used another IBM System Storage DS8000 with copy services functions enabled as the secondary storage subsystem. Both subsystems were connected via fiber paths.

We performed periodic failover/failback scenarios to ensure that application is consistent on the secondary volumes. We also tested disaster recovery support with failover/failback capability for the DS8000s.

Installing and setting up TPC-R

There are a number of prerequisites for the IBM TotalStorage Productivity Center for Replication for System z. For a complete list, see the **Installing > Prerequisites** section in the TPC-R Information Center at publib.boulder.ibm.com/infocenter/tivihelp/v4r1/index.jsp?topic=/com.ibm.rm33.doc/welcome.html.

TPC-R V3.3 will not install into a WebSphere Application Server network deployment (ND) configuration; it will only install properly into a base server setup. Also, there should be only a single copy of the application running within a single sysplex.

The product install will update the product's SMP/E installed files as part of the customization. In our case, the SMP/E build is performed on a separate sysplex and the product code is copied to our zPET systems where we will actually run it.

On our target system, we did the following:

- Mounted the TPC-R SMP/E copied file system in read/write mode. Normally, we would prefer to have our SMP/E product code mounted as read-only.
- Re-ran the setup jobs on the target system to customize the copied service update.
- We used a symbolic link to point to the current copy of the TPC-R service, rather than over-writing or replacing it on our target systems. This allows us to mount each service level at different directories,

Because the TPC-R customization jobs create directories and copy files into the WebSphere Application Server V6.1 configuration files, we made sure the jobs had access to the WebSphere Application Server directories as well as the TPC-R files. We ran the jobs under a user ID with UID=0 and other BPX.FILEATTR.* authorities. See the program directory for full requirements.

TPC-R also keeps various logging files within the WebSphere Application Server directories. These logs can add 200M bytes or more to the WebSphere Application Server file system. We checked the WebSphere Application Server file system to make sure we had enough space available and potentially some room for growth for when debugging was enabled.

We checked the output of the customization jobs carefully, including the z/OS UNIX files created (install_RM.log and install_RM_err.log) for errors.

TPC-R product documentation

See the TPC-R home page at www-306.ibm.com/software/tivoli/products/totalstorage-replication/ for full information about the product. The home page contains links to detailed information, in particular, the TPC-R Information Center and support.

Our workloads

We run a variety of workloads in our pseudo-production environment. Our workloads are similar to those that our customers use. In processing these workloads, we perform many of the same tasks as customer system programmers. Our goal, like yours, is to have our workloads up 24 hours a day, 7 days a week (24 x 7). We have workloads that exercise the sysplex, networking, and application enablement characteristics of our configuration.

Table 8 summarizes the workloads we run during our prime shift and off shift. We describe each workload in more detail below.

Table 8. Summary of our workloads

Shift	Base system workloads	Application enablement workloads	Networking workloads	Database product workloads
Prime shift	• Automatic tape switching • Batch pipes • JES2/JES3 printer simulators	• Enterprise Identity Mapping (EIM) • IBM HTTP Server • LDAP Server • Kerberos Server • z/OS UNIX Shelltest (rlogin/telnet) • z/OS UNIX Shelltest (TSO) • WebSphere Application Server for z/OS • WebSphere MQ for z/OS • WebSphere Message Broker	• AutoWEB • FTP workloads • MMFACTS for NFS • NFSWL • Silk Test NFS video stream • TCP/IP CICS sockets • TN3270	• CICS DBCTL • CICS/DB2 • CICS/QMF online queries • CICS/RLS batch • CICS/RLS online • CICS/NRLS batch • CICS/NRLS online • DB2 Connect™ • DB2 online reorganization • DB2/RRS stored procedure • IMS AJS • IMS/DB2 • IMS full function • IMS SMQ fast path • QMF™ batch queries
Off shift	• Random batch • Automatic tape switching • JES2/JES3 printer simulators	• Enterprise Identity Mapping (EIM) • IBM HTTP Server • LDAP Server • Kerberos Server • z/OS UNIX Shelltest (rlogin/telnet) • z/OS UNIX Shelltest (TSO) • WebSphere Application Server for z/OS • WebSphere MQ for z/OS • WebSphere Message Broker	• FTP workloads • Silk Test NFS video stream • MMFACTS for NFS	• CICS /DBCTL • CICS/DB2 • CICS/RLS batch • CICS RLS online • CICS/NRLS batch • CICS/NRLS online • DB2 DDF • DB2 utility • IMS/DB2 • IMS utility • MQ/DB2 bookstore application • QMF online queries

Base system workloads

We run the following z/OS base (MVS) workloads:

BatchPipes®: This is a multi-system batch workload using BatchPipes. It drives high CP utilization of the coupling facility.

Automatic tape switching: We run 2 batch workloads to exploit automatic tape switching and the ATS STAR tape sharing function. These workloads use the Virtual Tape Server and DFSMSrmm™, as described in our December 1998 test report, and consist of DSSCOPY jobs and DSSDUMP jobs. The DSSCOPY jobs copy particular data sets to tape, while the DSSDUMP jobs copy an entire DASD volume to tape.

Both workloads are set up to run under Tivoli Workload Scheduler (TWS, formerly called OPC) so that 3 to 5 job streams with hundreds of jobs are all running at the same time to all systems in the sysplex. With WLM-managed initiators, there are no system affinities, so any job can run on any system. In this way we truly exploit the capabilities of automatic tape switching.

Tivoli Workload Scheduler (TWS) EXIT 51 tip

Due to changes in JES2 for z/OS V1R7, TWS has made a new EXIT called EXIT51. TWS will only support TWS 8.1 or higher for z/OS V1R7 users. If you have z/OS V1R7 and use TWS 8.1 or higher you will need to:

- compile and linkedit your usual JES2/TWS EXITS
- compile and linkedit the new EXIT51.

EQQXIT51 is provided in the SEQQSAMP Lib. You will also need to add the following to both your JES2 PARM and existing OPCAXIT7 statement:

```
LOAD(TWSXIT51)
EXIT(51) ROUTINES=TWSENT51,STATUS=ENABLED
```

Once EXIT51 was installed and enabled we found no problems with our normal use of TWS 8.1.

JES2/JES3 printer simulators: This workload uses the sample functional subsystem (FSS) and the FSS application (FSA) functions for JES2 and JES3 output processing.

Random batch: This workload is a collection of MVS test cases that invoke many of the functions (both old and new) provided by MVS.

Application enablement workloads

We run the following application enablement workloads:

Enterprise Identity Mapping (EIM)

This workload exercises the z/OS EIM client and z/OS EIM domain controller. It consists of a shell script running on a z/OS image that simulates a user running EIM transactions.

HFS/zFS FILESYSTEM RECURSIVE COPY/DELETE

This TPNS driven workload copies over 700 directories from one large filesystem to another. It then deletes all directories in the copy with multiple remove (rm) commands.

IBM HTTP Server

These workloads are driven from AIX/RISC workstations. They run against various HTTP server environments, including the following:

- HTTP scalable server
- HTTP standalone server
- Sysplex distributor routing to various HTTP servers

These workloads access the following:
- MVS datasets
- FastCGI programs
- Counters
- Static html pages
- Static pages through an SSL connection
- REXX Exec through GWAPI
- Protection through RACF userid
- Sysplex Distributor
- Standalone http server
- Scalable http server

ICSF
This workload runs on MVS. It is run by submitting a job through TSO. This one job kicks off 200+ other jobs. These jobs are set up to use ICSF services to access the crypto hardware available on the system. The goal is to keep these jobs running 24/7.

LDAP Server
LDAP Server consists of the following workloads:

- Segue Silk Performer - is setup on a remote Windows machine. The workload is setup to run a Performer Script for 20 users. The script is designed to issue several LDAP commands (ldapsearch, ldapadd, ldapdelete) issued to the z/OS LDAP server. At the start of the workload simulation, each virtual user is setup to have a 15 second delay between executing the script, thus making the simulation more ″customer like″. This workload simulation is then executed on a 24/7 basis.
- Tivoli Access Manager - Tivoli Access Manager uses z/OS LDAP to store user information. The workload that is executed is a shell script that consists of several TAM user admin commands that places stress on the TAM/LDAP environment.
- Mindcraft Workload Simulator - The DirectoryMark benchmark is designed to measure the performance of server products that use LDAP, We have this product installed on a Windows server machine. Scripts generated by DirectoryMark are run against z/OS LDAP on a 24/7 basis.
- Authentication - This workload is driven from an AIX/RISC workstation. It runs against the IBM HTTP Server on z/OS and Apache on Linux to provide LDAP authentication when accessing protected resources.

NAS (kerberos)
This workload runs from the shell as a shell script. It uses both the z/OS LDAP and z/OS EIM client to bind through kerberos with EIM and LDAP.

z/OS UNIX Shelltest (rlogin/telnet)
In this workload, users log in remotely from an RS/6000® workstation to the z/OS shell using either rlogin or telnet and then issue commands.

z/OS UNIX Shelltest (TSO)
In this workload, simulated users driven by the Teleprocessing Network Simulator (TPNS) logon to TSO/E and invoke the z/OS UNIX shell and issue various commands. The users perform tasks that simulate real z/OS UNIX users daily jobs, for example:
- Moving data between the HFS and MVS data sets.
- Compiling C programs.
- Running shell programs.

WebSphere Application Server for z/OS

We run a number of different Web application workloads in our test environment on z/OS. Generally, each workload drives HTTP requests to Web applications that consist of any combination of static content (such as HTML documents and images files), Java™ Servlets, JSP pages, and Enterprise JavaBeans™ (EJB) components. These Web applications use various connectors to access data in our DB2, CICS, or IMS subsystems.

Our Web application workloads currently include the following:

- J2EE applications (including persistent (CMP and BMP) and stateless session EJB components) that:
  - Access DB2 using JDBC
  - Access CICS using the CICS Common Client Interface (CCI)
  - Access IMS using the IMS Connector for Java CCI
  - Access WebSphere MQ using Java Message Service (JMS)
  - Access Websphere MQ and the Websphere Message Broker
- Non-J2EE applications (only static resources, Servlets, and JSP pages) that:
  - Access DB2 using JDBC
  - Access CICS using CICS CTG
  - Access IMS using IMS Connect
- Other variations of the above applications, including those that:
  - Access secure HTTPS connections using SSL
  - Perform basic mode authentication
  - Use HTTP session data
  - Use connection pooling
  - Use persistent messaging
  - Use RACF or LDAP for Local OS security
  - Use WebSphere Network Deployment (ND) configuration(s)
  - Utilize Sysplex Distributor
  - Use HTTP Server / J2EE Server clustering
  - Use DB2 Legacy RRS / DB2 UDB JCC driver(s)

WebSphere MQ for z/OS workloads

Our WebSphere MQ environment includes one WebSphere MQ for z/OS queue manager on each system in the sysplex. We have two queue sharing groups: one with three queue managers and another with four queue managers.

Our workloads test the following WebSphere MQ features:

- CICS Bridge
- IMS Bridge
- Distributed queueing with SSL and TCP/IP channels
- Large messages
- Shared queues
- Clustering
- Transaction coordination with RRS
- CICS Adapter

We use the following methods to drive our workloads (not all workloads use each method):

- Batch jobs
- Web applications driven by WebSphere Studio Workload Simulator
- TPNS TSO users running Java programs through z/OS UNIX shell scripts

Some of the workloads that use WebSphere MQ for z/OS include the following:

MQ batch stress for non-shared queues: This workload runs on one system and stresses WebSphere MQ for z/OS by issuing MQI calls. These calls include a variety of commands affecting local queues.

MQ batch stress for shared queues: This workload runs on one system and stresses WebSphere MQ for z/OS by issuing MQI calls. These calls include a variety of commands affecting shared queues. Workload parameters control the number of each type of call.

DQM and DQMssl: This workload tests the communication between z/OS queue managers using SSL TCPIP channels. The application puts messages on remote queues and waits for replies on its local queues.

MQCICS: This workload uses the MQ CICS bridge to run a transaction that updates a DB2 parts table. The CICS bridge request and reply queues are local queues that have persistent messages. We also have a non-Web version of MQCICS that uses shared cluster queues with persistent messages. We defined a separate coupling facility structure for this application. Another version of the workload uses the MQ CICS adapter to process transactions. All three queues (request, reply, and initiation) are shared. All members of our queue sharing group have a CICS region monitoring the queue.

MQLarge: This workload tests various large message sizes by creating temporary dynamic queues and putting large messages on those queues. Message sizes vary from 1MB to 100MB starting in increments of 10MB. The script running the application randomly chooses a message size and passes this to the mqLarge program. mqLarge then dynamically defines a queue using model queues that have their maxmsgl set to accommodate the message.

WebSphere Message Broker workloads

Our WebSphere Message Broker environment consists of six message brokers: three on test systems and three on production systems. All are running Websphere Message Broker v6.0. We will refer to this broker version as WMB. We use the following methods to drive our workloads (not all workloads use each method):

- Web applications driven by WebSphere Studio Workload Simulator
- Batch jobs
- TPNS TSO users running Java programs through z/OS UNIX shell scripts

The Web applications consist of html pages, java servlets, and message flows to process the messages. These Java-based workloads have recently been converted to use Websphere Application Server 5.1 instead of the IBM HTTP Server with the WebSphere V4.0 plugin.

Retail_IMS: This workload tests message manipulation by taking a message, extracting certain fields from it, and adding an IMS header.

Retail_Info: This workload tests inserting and deleting fields from a message into a simple DB2 table.

Retail_Wh: This workload tests inserting and deleting an entire message (using a data warehouse node) into a LOB DB2 table.

We have two batch-driven workloads:

Sniffer: This workload tests basic MQ and broker functionality using persistent and non-persistent messages. It is based on SupportPac™ IP13: Sniff test and Performance on z/OS. (See http://www-306.ibm.com/software/integration/support/supportpacs/category.html#cat1)

Football: This workload tests basic broker publish/subscribe functionality. Using the Subscribe portion of the workload, a subscription is registered with the broker. The Publish portion publishes messages to the broker, which then routes them to the matching subscribers. Like the Sniffer workload, this workload is based on SupportPac IP13.

We have one TPNS workload that uses WMB:

Retail_TPNS: This workload is another version of Retail_IMS, but rather than being driven by WebSphere Studio Workload Simulator, it is driven by TPNS through z/OS UNIX shell scripts.

Networking workloads

We run the following networking workloads:

FTP workloads:

- **FTPHFS/DB2:** This client/server workload simulates SQL/DB2 queries through an FTP client.
- **FTPHFS(Linux):** This workload simulates users logging onto a Linux client through telnet or FTP and simulates workloads between the z/OS servers and the LINUX client.
- **FTP TPNS:** This workload uses TPNS to simulate FTP client connections to the z/OS server.
- **FTPWL:** This client/server workload automates Linux clients performing FTP file transfers across Token Ring and Ethernet networks. This workload also exercises the z/OS Domain Name System (DNS). Files that are transferred reside in both z/OS HFS and MVS non-VSAM data sets. Future enhancements to this workload will exploit the z/OS workload manager DNS.

MMFACTS for NFS: This client/server workload is designed to simulate the delivery of multimedia data streams, such as video, across the network. It moves large volumes of randomly-generated data in a continuous, real-time stream from the server (in our case, z/OS) to the client. Data files can range in size from 4 MB to 2 Gigabytes. A variety of options allow for variations in such things as frame size and required delivery rates.

NFSWL: This client/server workload consists of shell scripts that run on our AIX® clients. The shell script implements reads, writes, and deletes on an NFS mounted file system. We mount both HFS and zFS file systems that reside on z/OS. This workload is managed by a front end Web interface.

AutoWEB: This client/server workload is designed to simulate a user working from a Web Browser. It uses the following HTML meta-statement to automate the loading of a new page after the refresh timer expires:

```
<meta http-equiv='Refresh' content='10; url=file:///filename.ext'>
```

This workload can drive any file server, such as LAN Server or NFS. It also can drive a Web Server by changing the URL from `url=file:///filename.ext` to `url=http://host/filename.ext`.

Silk Test NFS video stream: This client/server workload is very similar to that of MMFACTS except that it sends actual video streams across the network instead of simulating them.

TCP/IP CICS sockets: This TPNS workload exercises TCP/IP CICS sockets to simulate real transactions.

TN3270: This workload uses TPNS to simulate TN3270 clients which logon to TSO using generic resources. This workload exploits Sysplex Distributor.

Database product workloads

Our database product workloads include online transaction processing (OLTP) workloads, batch workloads, and our WebSphere MQ / DB2 bookstore application.

Database product OLTP workloads

Our sysplex OLTP workloads are our mission critical, primary production workloads. Each of our 3 application groups runs different OLTP workloads using CICS or IMS as the transaction manager:

- Application group 1 — IMS data sharing, including IMS shared message queue
- Application group 2 — VSAM record level sharing (RLS) and non-RLS
- Application group 3 — DB2 data sharing (four different OLTP workloads, as well as several batch workloads).

Note that our OLTP workloads, which are COBOL, FORTRAN, PL1, or C/C++ programs, are Language Environment® enabled (that is, they invoke Language Environment support).

IMS data sharing workloads: In application group one, we run three IMS data sharing workloads:

- CICS/DBCTL
- IMS EMHQ Fast Path
- IMS SMQ full function
- IMS automated job submission (AJS)

Highlights of our IMS data sharing workloads include:

- Full function, Fast Path, and mixed mode transactions
- Use of virtual storage option (VSO), shared sequential dependent (SDEP) databases, generic resources, and High Availability Large Databases (HALDB)
- Integrity checking on INSERT calls using SDEP journaling
- A batch message processing (BMP) application to do integrity checking on REPLACE calls
- A set of automatically-submitted BMP jobs to exercise the High-Speed Sequential Processing (HSSP) function of Fast Path and the reorg and SDEP scan and delete utilities. This workload continuously submits jobs at specific intervals to run concurrently with the online system. We enhanced this workload based on customer experiences to more closely resemble a real-world environment.

VSAM/RLS data sharing workload: In application group 2, we run one OLTP VSAM/RLS data sharing workload. This workload runs transactions that simulate a banking application (ATM and teller transactions). The workload also runs transactions that are similar to the IMS data sharing workload that runs in application group 1, except that these transactions use VSAM files.

VSAM/NRLS workload: Also in application group 2, we added two new workloads. One uses transactions similar to our VSAM/RLS workload but

accessing VSAM non-RLS files. The other is a very I/O-intensive workload that simulates a financial brokerage application.

DB2 data sharing workloads: In application group 3, we run four different DB2 data sharing OLTP workloads. These workloads are also similar to the IMS data sharing workload running in application group 1.

In the first of the DB2 workloads, we execute 8 different types of transactions in a CICS/DB2 environment. This workload uses databases with simple and partitioned table spaces.

In the second of our DB2 workloads, we use the same CICS regions and the same DB2 data sharing members. However, we use different transactions and different databases. The table space layout is also different for the databases used by the second DB2 workload—it has partitioned table spaces, segmented table spaces, simple table spaces, and partitioned indexes.

Our third workload is a derivative of the second, but incorporates large objects (LOBs), triggers, user defined functions (UDFs), identity columns, and global temporary tables.

The fourth workload uses IMS/TM executing 12 different transaction types accessing DB2 tables with LOBs. It also excercises UDFs, stored procedures and global temporary tables.

Database product batch workloads

We run various batch workloads in our environment, some of which we will describe here. They include:

- IMS Utility
- RLS batch (read-only) and TVS batch
- DB2 batch workloads

We run our batch workloads under TWS control and use WLM-managed initiators. Our implementation of WLM batch management is described in our December 1997 test report.

DB2 batch workloads: Our DB2 batch workloads include:

- DB2 Online reorganization
- DB2/RRS stored procedure
- QMF batch queries
- DB2 utilities
- DB2 DDF

Our DB2 batch workload has close to 2000 jobs that are scheduled using TWS, so that the jobs run in a certain sequence based on their inter-job dependencies.

WebSphere MQ / DB2 bookstore application

Our multi-platform bookstore application lets users order books or maintain inventory. The user interface runs on AIX, and we have data in DB2 databases on AIX and z/OS systems. We use WebSphere MQ for z/OS to bridge the platforms and MQ clustering to give the application access to any queue manager in the cluster. See our December 2001 test report for details on how we set up this application.

Chapter 2. About our networking environment

This chapter describes our networking environment, including a high-level overview of our TCP/IP, network file systems, and VTAM configurations and workloads.

Our networking configuration

Figure 3 provides a logical view of our networking configuration.

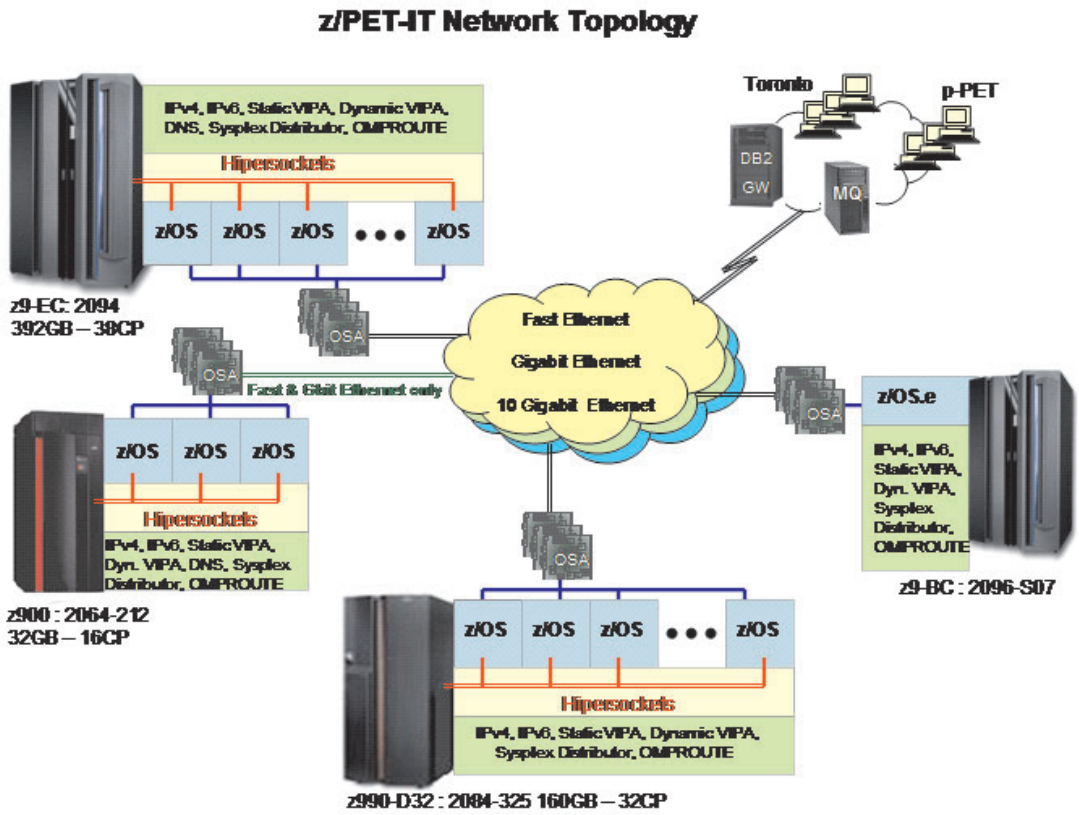


Figure 3. Our networking topology

Configuration overview

Our networking environment is entirely Ethernet. Currently we have Fast Ethernet, Gigabit Ethernet, and 10 Gigabit Ethernet, each running on separate networks. This setup provides a robust environment for our z/OS testing. Across these networks we run workloads that exercise many z/OS components and IBM products.

The following describes what is illustrated in Figure 3.

- We have OSA Fast Ethernet, OSA Gigabit Ethernet, and OSA 10 Gigabit Ethernet configured on three of our 4 CECs. Since our fourth CEC, the z900, does not support the 10 Gigabit OSA feature, we only have the OSA Fast Ethernet and OSA Gigabit Ethernet features configured.

- We use OMPROUTE on each z/OS image to provide dynamic OSPF routing support across our data center.
- We have a DNS setup using master and slave all on z/OS.
- We have dynamic XCF configured so that we can use hypersockets on the CEC's where there is more than one image for communications between those images.
- All of the networks are VLAN Tagged.
- We have a fully implemented IPv6 environment.
- We run many sysplex distributors for workload balancing with a variety of distribution methods using IPv4 and IPv6.

Our IPv6 environment configuration

We currently run a fully implemented IPv6 environment utilizing IPv6 OMPROUTE and DVIPA/Sysplex distributor, This is used to support WebSphere MQ V6 and DB2 V9.1 implementations.

z/OS UNIX System Services changes and additions

The following are the changes and additions we made to z/OS UNIX System Services:

1. Changing BPXPRMxx to add IPv6 support

   We made the following changes to BPXPRMxx to add IPv6 support:

```
NETWORK DOMAINNAME(AF_INET6)
 DOMAINNUMBER(19)
 MAXSOCKETS(60000)
 TYPE(INET)
```

   Note: INADDRANYPORT and INADDRANYCOUNT values are used for both IPv4 and IPv6 when the BPXPRMxx is configured for both IPv4 and IPv6 support. If AF_INET is specified, it is ignored and the values from the NETWORK statement for AF_INET are used if provided. Otherwise, the default values are used.

2. Adding NETWORK statements to have a TCP/IP stack that supports IPv4 and IPv6.

   We added the following two NETWORK statements to have a TCP/IP stack that supports IPv4 and IPv6:

```
 FILESYSTYPE TYPE(CINET) ENTRYPOINT(BPXTCINT)
 NETWORK DOMAINNAME(AF_INET)
 DOMAINNUMBER(2)
 MAXSOCKETS(2000)
 TYPE(CINET)
 INADDRANYPORT(20000)
 INADDRANYCOUNT(100)
NETWORK DOMAINNAME(AF_INET6)
 DOMAINNUMBER(19)
MAXSOCKETS(3000)
 TYPE(CINET)
SUBFILESYSTYPE NAME(TCPCS) TYPE(CINET) ENTRYPOINT(EZBPFINI)
SUBFILESYSTYPE NAME(TCPCS2) TYPE(CINET) ENTRYPOINT(EZBPFINI)
SUBFILESYSTYPE NAME(TCPCS3) TYPE(CINET) ENTRYPOINT(EZBPFINI)
```

TCPIP Profile changes

We made the following additions to our IPv6 INTERFACE statements:

```
INTERFACE OSA9E0V6
DEFINE IPAQENET6
 PORTNAME GBPRT9E0
 IPADDR FEC0:0:0:1:x:xx:xx:xxx ;(Site-Local Address)
 3FFE:0302:0011:2:x:xx:xx:xxx ; (Global Address)
```

Note: In order to configure a single physical device for both IPv4 and IPv6 traffic, you must use DEVICE/LINK/HOME for the IPv4 definition and INTERFACE for the IPv6 definition, so that the PORTNAME value on the INTERFACE statement matches the device_name on the DEVICE statement.

Dynamic XCF addition

We made the following addition for our Dynamic XCF:

```
IPCONFIG6 DYNAMICXCF FEC0:0:0:1:0:168:49:44
```

Dynamic VIPA additions

The following statement was added to our VIAPDYNAMIC section:

```
VIPADEFINE V6Z2FTP 2003:0DB3:1::2
VIPADISTRIBUTE SYSPLEXPORTS V6Z2FTP PORT 20 21
 DESTIP FEC0:0:0:1:0:168:49:37
```

Note: V6Z2FTP is the INTERFACE name for this VIPA.

OMPROUTE addition

Setting up OMPROUTE only requires adding the INTERFACE name to the OMPROUTE profile for the basic setup that we used.

```
IPV6_OSPF_INTERFACE
 Name = OSA9E0V6;
```

Note: During testing we encountered the following message:

EZZ7954I IPv6 OSPF adjacency failure, neighbor 192.168.25.33, old state 128, new state 4, event 10

The neighbor id in the message is the ROUTERID from the OMPROUTE profile. It will not show an IPv6 address.

NAMESERVER changes

We created separate IPv6 names for each LPAR. To keep things simple for the system name, we used the existing LPAR name with IP6 as the suffix. For the IPv6 ip addresses, we used a common prefix and used the IPv4 address as the suffix. This made it easier to identify for diagnosing problems.

Forward file changes

The following change was made to our forward file:

```
J80IP6 IN AAAA 3FFE:302:11:2:9:12:20:150
```

Reverse file entry addition: We added the following for the reverse file entry:

```
$TTL 86400
 $ORIGIN 2.0.0.0.1.1.0.0.2.0.3.0.E.F.F.3.IP6.ARPA.
 @  IN SOA Z0EIP.PDL.POK.IBM.COM. ALEXSA@PK705VMA
    ( 012204 ;DATE OF LAST CHANGE TO THIS FILE
   21600 ;REFRESH VALUE FOR SECONDARY NS (IN SECS) 1800 ;
   RETRY VALUE FOR SECONDARY NS (IN SECS)
   48384 ;EXPIRE DATA WHEN REFRESH NOT AVAILABLE
   86400 ) ;MINIMUM TIME TO LIVE VALUE (SECS)
@   IN NS Z0EIP.PDL.POK.IBM.COM. ; PRIMARY DNS
0.5.1.0.0.2.0.0.2.1.0.0.9.0.0.0 IN PTR  J80IP6.PDL.POK.IBM.COM.
```

Comparing the network file systems

If you are a faithful reader of our test report, you might have noticed that we have changed our Network File System (NFS) approach a number of times, depending on the circumstances at the moment. Currently, we have the z/OS NFS (called DFSMS/MVS® NFS in OS/390 releases prior to R6) on system Z0.

NFS allows files to be transferred between the server and the workstation clients. To the clients, the data appears to reside on a workstation fixed disk, but it actually resides on the z/OS server.

With z/OS NFS, data that resides on the server for use by the workstation clients can be either of the following:

- z/OS UNIX files that are in a hierarchical file system (HFS). The z/OS NFS is the only NFS that can access files in an HFS. You need to have z/OS NFS on the same system as z/OS UNIX and its HFS if you want to use the NFS to access files in the HFS.
- Regular MVS data sets such as PS, VSAM, PDSs, PDSEs, sequential data striping, or direct access.

Migrating to the z/OS NFS: We plan to implement some of the new functions available in z/OS NFS, such as file locking over the z/OS NFS server and file extension mapping support. You can read descriptions of these new functions in *z/OS Network File System Guide and Reference*, SC26-7417. In addition, you can read about WebNFS support in our December 1999 test report at *OS/390 Parallel Sysplex Test Report*, and the use of the LAN Server NFS in our December 2004 edition at *zSeries Platform Test Report*. All of our editions can be found at:

`http://www.ibm.com/servers/eserver/zseries/zos/integtst/library.html`

Our VTAM configuration

Figure 4 illustrates our VTAM® configuration.

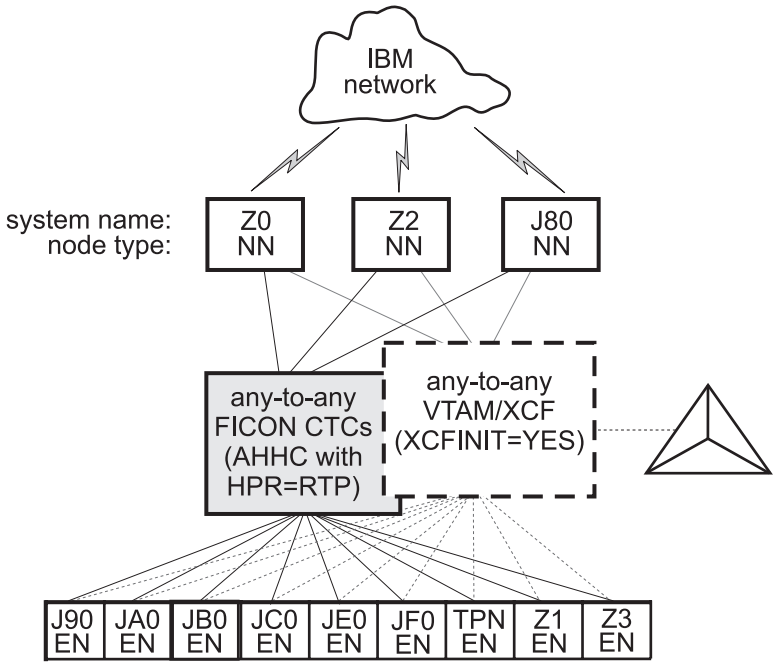


Figure 4. Our VTAM configuration

TPNS runs on our system TPN and routes CICS logons to any of the other systems in the sysplex.

Our VTAM configuration is a pure any-to-any AHHC. Systems Z0, Z2, and J80 are the network nodes (NNs) and the remaining systems are end nodes (ENs).

We also have any-to-any communication using XCF signalling, where XCF can use either CTCs, coupling facility structures, or both. This is called dynamic definition of VTAM-to-VTAM connections.

We are configured to use both AHHC and XCF signalling for test purposes.

Networking workloads

For information about our networking workloads, see "Our workloads" on page 14.

Enabling NFS recovery for system outages

In z/OS V1R6, we improved NFS recoverability and availability by using Automatic Restart Management (ARM) and dynamic virtual IP address (DVIPA) with our NFS server. With these enhancements, the NFS server is automatically moved to another MVS image in the sysplex during a system outage.

Note: We are running a shared HFS environment.

We used the following documentation to help us implement ARM for NFS recovery.

- Automatic Restart Management
  - ARMWRAP as described in the IBM Redpaper *z/OS Automatic Restart Manager* available on the IBM Redbooks Web site.
  - *z/OS MVS Setting Up a Sysplex*, SA22-7625
- Dynamic VIPA(DVIPA)
  - *z/OS Communications Server: IP Configuration Guide*, SC31-8775

Setting up the NFS environment for ARM and DVIPA

Part 1 of Figure 5 on page 28: illustrates how the NFS server on MVS A acquires DVIPA 123.456.11.22. The AIX clients issue a hard mount specifying DVIPA 123.456.11.22. Before the enhancements, the AIX clients specified a static IP address for MVS A. A system outage would result in the mounted file systems being unavailable from the AIX client's perspective until MVS A was restarted.

Part 2 of Figure 5 on page 28 : illustrates that when an outage of MVS A occurs, ARM automatically moves the NFS server to MVS B. The NFS Server on MVS B acquires the DVIPA 123.456.11.22. From the AIX client's perspective the mounted file systems become available once the NFS server has successfully restarted on MVS B. The original hard mount persists.

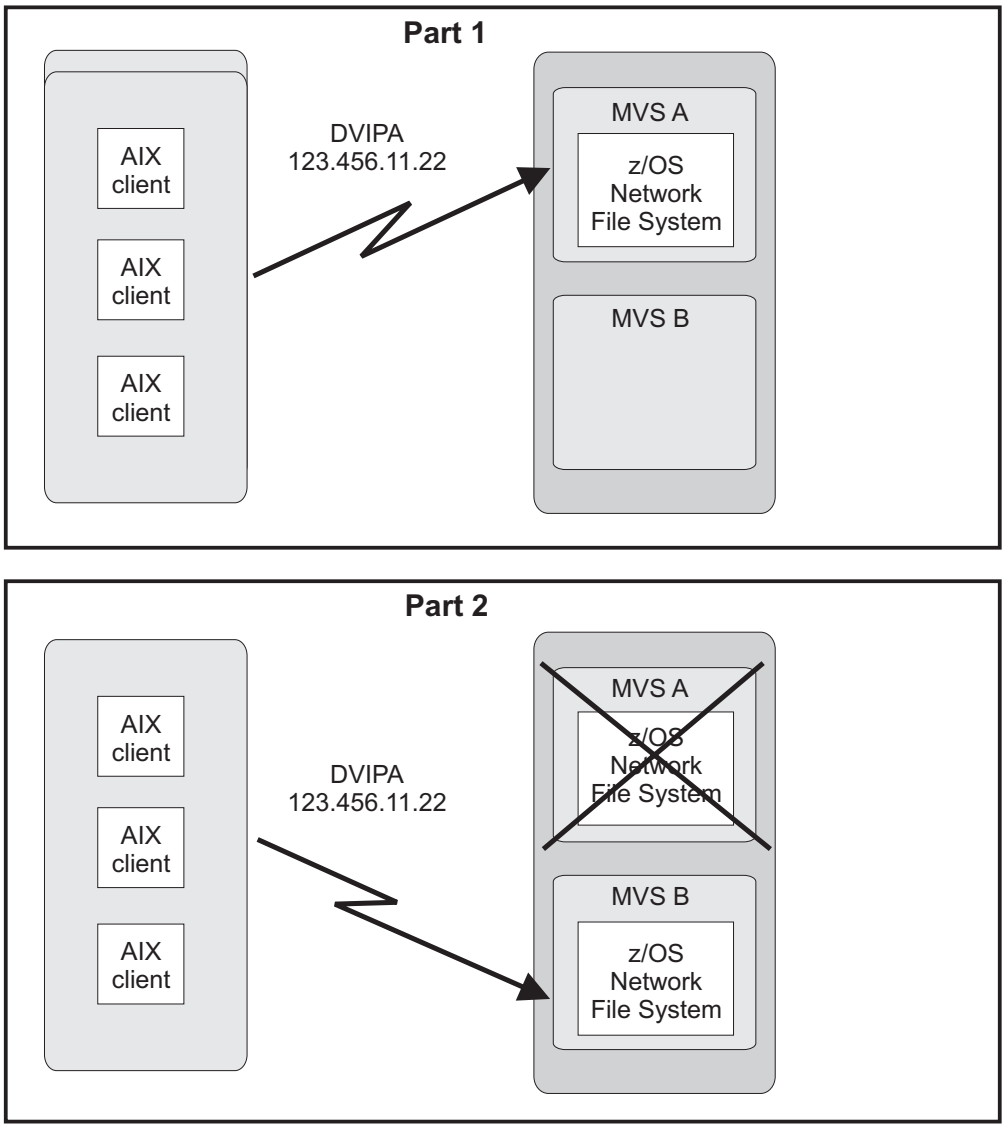


Figure 5. NFS configuration

Note: An ARM enabled NFS will not automatically move back to MVS A after MVS A recovers.

Step for setting up our NFS environment

We performed the following steps to set up our NFS environment for ARM and DVIPA:

1. Acquiring dynamic VIPA:

   We added the following statement in the TCP/IP profiles for MVSA and MVSB to allow NFS to acquire dynamic VIPA:

   ```
   VIPARANGE DEFINE 255.255.255.255 123.456.11.22 ; NFS VIPA
   ```

   We recycled TCPIP on MVSA and MVSB to activate the above changes.

   Note: You could also use the `VARY TCPIP,,OBEYFILE` command with a data set that contains VIPARANGE statement.

   ---

2. Defining the NFS element:

We added the following statement to our ARM policy member (ARMPOLxx) in SYS.PARMLIB member to define the NFS element:

```
RESTART_GROUP(NFSGRP)
  TARGET_SYSTEM(MVSB)
  FREE_CSA(600,600)
    ELEMENT(NFSSELEM)
      RESTART_ATTEMPTS(3,300)
      RESTART_TIMEOUT(900)
      READY_TIMEOUT(900)
```

---

3. Loading the ARM policy:

   We ran the IXCMIAPU utility to load ARMPOLxx and then activated the policy:

   ```
   setxcf start,policy,type=arm,polname=armpolxx
   ```

---

4. Registering NFS using an ARM policy:

   We used ARMWRAP, the ARM JCL Wrapper with the following parameters to register NFS as ARM element:

   ```
   //*******************************************************************
   //*REGISTER ELEMENT 'NFSSELEM' ELEMENT TYPE 'SYSTCPIP' WITH ARM
   //*REQUIRES ACCESS TO SAF FACILITY IXCARM.SYSTCPIP.NFSSELEM
   //ARMREG EXEC PGM=ARMWRAP,
   //       PARM=('REQUEST=REGISTER,READYBYMSG=N,',
   //             'TERMTYPE=ALLTERM,ELEMENT=NFSSELEM,',
   //             'ELEMTYPE=SYSTCPIP')
   //* ---------------------------------------------------------------- *
   //* DELETE VIPA FOR NFS SERVER                                       *
   //* ---------------------------------------------------------------- *
   //DELVIPA  EXEC PGM=EZBXFDVP,
   //            PARM='POSIX(ON) ALL31(ON) /-p TCPIP -d &VIPA'
   //SYSPRINT DD SYSOUT=*
   //* ---------------------------------------------------------------- *
   //* ACQUIRE VIPA FOR NFS SERVER                                      *
   //* ---------------------------------------------------------------- *
   //DEFVIPA  EXEC PGM=EZBXFDVP,
   //            PARM='POSIX(ON) ALL31(ON) /-p TCPIP -c &VIPA'
   //SYSPRINT DD SYSOUT=*
   ```

---

5. Terminating the address space:

   The following example shows what is executed when the address space is terminated:

   ```
   //* ---------------------------------------------------------------- *
   //* DELETE VIPA FOR NFS SERVER                                       *
   //* ---------------------------------------------------------------- *
   //DELVIPA  EXEC PGM=EZBXFDVP,
   //            PARM='POSIX(ON) ALL31(ON) /-p TCPIP -d &VIPA'
   //SYSPRINT DD SYSOUT=*
   //*******************************************************************
   //*FOR NORMAL TERMINATION,DEREGISTER FROM ARM
   //*FOR NORMAL TERMINATION,DEREGISTER FROM ARM
   //*********************************************
   //ARMDREG EXEC PGM=ARMWRAP,
   //        PARM=('REQUEST=DEREGISTER')
   ```

Chapter 3. About our security environment

Information about our security computing environment includes:

- "Our Integrated Cryptographic Service Facility (ICSF) configuration"
- "Using Network Authentication Service (Kerberos)" on page 32
- "Using LDAP Server" on page 40
- "Using System SSL" on page 46
- "Cryptographic Services PKI Services updates" on page 46
- "Implementing and using PKCS #11 support" on page 47
- "Implementing and using the RACF Java API" on page 48

Our Integrated Cryptographic Service Facility (ICSF) configuration

z/OS Integrated Cryptographic Service Facility (ICSF) is a software element of z/OS that works with the hardware cryptographic features and the Security Server (RACF) to provide secure, high-speed cryptographic services in the z/OS environment. ICSF provides the application programming interfaces by which applications request the cryptographic services. The cryptographic feature is secure, high-speed hardware that performs the actual cryptographic functions.

The available cryptographic hardware features are dependent on the server.

In our sysplex, we ran with ICSF, FMID HCR7740, which is available with z/OS V1R9. Because we have many types of servers in our environment, we run with various cryptographic hardware features. Following is a list of cryptographic hardware features we currently have in our environment:

- Crypto Express2 Accelerator (CEX2A)
- Crypto Express2 Coprocessor (CEX2C)
- PCI Cryptographic Accelerator (PCICA)
- PCI X Cryptographic Coprocessor (PCIXCC)
- CP Assist for Cryptographic Functions (CPACF)
- CP Assist for Cryptographic Functions DES/TDES Enablement (CPACF, feature 3863)
- PCI Cryptographic Coprocessor (PCICC)
- Cryptographic Coprocessor Facility (CCF)

Since our goal is to run a customer-like environment, we have various workloads and jobs which take advantage of the products that interface with ICSF (which interfaces with the cryptographic hardware). These products include the following:

- SSL (through WebSphere Application Server, FTP, HTTP, LDAP and CICS)
- Enterprise Key Manager Offering for Tape Encryption
- Encryption Facility for z/OS V1 R1
- Encryption Facility for z/OS V1 R2
- Network Authentication Service (Kerberos) (through LDAP, EIM, and FTP)

We also have an ICSF specific workload that runs daily which exercises the cryptographic services available through the ICSF Callable Services.

Note: For additional information on the Enterprise Key Manager Offering for Tape Encryption and the Encryption Facility for z/OS V1 R2, see our June 2007 test report. For Encryption Facility for z/OS V1R1, see our our December 2006 test report.

Using Network Authentication Service (Kerberos)

Integrated Security Services Network Authentication Service for z/OS is the IBM z/OS program based on Kerberos Version 5 and GSS.

Network Authentication Service configuration and verification

Figure 6 shows an overview of our Network Authentication Service (NAS) configuration.

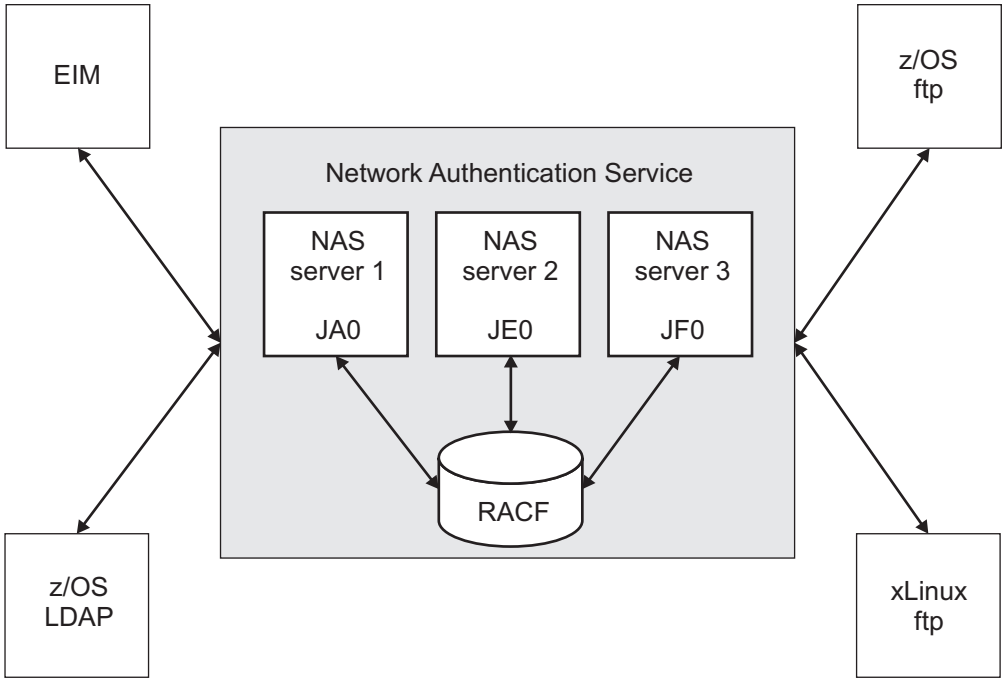


Figure 6. Overview of our Network Authentication Service configuration

We currently have three NAS servers configured within one sysplex. All of the servers use RACF as the registry database.

The EIM, z/OS LDAP, z/OS ftp and xLinux ftp clients have all been enabled to exploit NAS, as discussed in previous editions of our test report:

- For information about our enablement of EIM with NAS, see our September 2004 test report, *zSeries Platform Test Report Version 1 Release 6*, SA22-7997-00.
- For information about our enablement of z/OS LDAP with NAS, see our December 2002 test report, *Parallel Sysplex Test Report Version 1 Release 3 & Version 1 Release 4*, SA22-7663-07.
- For information about our enablement of z/OS ftp and xLinux ftp with NAS, see our December 2005 test report, *zSeries Platform Test Report for z/OS and Linux Virtual Servers Version 1 Release 7*, SA22-7997-02.

skrb5.conf file
Our skrb5.conf file follows the example provided in /usr/lpp/skrb/examples/skrb5.conf, except that we have configured for all encryption levels.

envar file
Our envar file follows the example provided in /usr/lpp/skrb.examples/skrbkdc.envar, except that we have configured for all encryption levels.

To validate our configuration, a **kinit** command is first issued to obtain our Kerberos credentials. Then, a transaction using each of the four clients is issued using those credentials.

Using AES encryption with Network Authentication Service

Network Authentication Service (NAS), has been enhanced in the z/OS V1R9 release to provide both AES128 and AES256 encryption levels. We'll talk about our experiences with enabling these encryption levels and then the exploitation of them.

We used the following documentation to help us with the enablement and exploitation:

- *z/OS Integrated Security Services Network Authentication Service Administration*, SC24-5926
- *z/OS Integrated Security Services LDAP Client Programming*, SC24-5924
- *z/OS Security Server RACF Command Language Reference*, SA22-7687
- *z/OS Integrated Security Services EIM Guide and Reference*, SA22-7875
- *z/OS Communications Server: IP Configuration Guide*, SC31-8775

Enabling AES encryption with Network Authentication Service

The NAS administration documentation indicates the following:

> Do not enable DES3, AES128 or AES256 ticket support until the Kerberos runtimes for all systems in the realm support that encryption type. Otherwise, you can obtain tickets that cannot be processed on a given system. In addition, do not enable DES3, AES128 or AES256 encryption support for user data unless all systems in the realm support that encryption type for user data. Otherwise, you can obtain session keys that are unusable for exchanging encrypted data. This means that all systems sharing the database must be running z/OS Version 1 Release 2 or later for DES3 or z/OS Version 1 Release 9 or later for AES128 or AES256.

Updating the /etc/krb5.conf file: A backup copy of each krb5.conf file was made with the following naming convention: krb5.conf.*date*, using the following command:

```
cp /etc/krb5.conf /etc/krb5.conf.date
```

The krb5.conf file located in /usr/lpp/skrb/examples was used as a guide when making the updates.

The default_tkt_enctypes and default_tgs_enctypes variables were updated to add the aes256-cts-hmac-sha1-96 and aes128-cts-hmac-sha1-96 encryption types.

```
default_tkt_enctypes = aes256-cts-hmac-sha1-96,aes128-cts-hmac-sha1-96,des3-cbc-sha1,
   des-hmac-sha1,des-cbc-md5,des-cbc-md4,des-cbc-crc
default_tgs_enctypes = aes256-cts-hmac-sha1-96,aes128-cts-hmac-sha1-96,des3-cbc-sha1,
   des-hmac-sha1,des-cbc-md5,des-cbc-md4,des-cbc-crc
```

It is important to remember that the encryption type selected is chosen from left to right. So, in our example, the first encryption type attempted would be aes256-cts-hmac-sha1-96.

As of the time of our testing, the NAS administration documentation makes it sound like all three variables, SKDC_TKT_ENCTYPES, default_tkt_enctypes, and default_tgs_enctypes are located in the krb5.conf file. This is not true. SKDC_TKT_ENCTYPES is located in the envar file. This will be clarified in a future edition of the documentation. The prologue for the sample krb5.conf file in /usr/lpp/skrb/examples does indicate that the SKDC_TKT_ENCTYPES is located in the envar file.

Updating the /etc/skrb/home/kdc/envar file: A backup copy of each envar file was made with the following naming convention: envar.*date*, using the following command:

```
cp /etc/skrb/home/kdc/envar /etc/skrb/home/kdc/envar.date
```

The envar file located in /usr/lpp/skrb/examples/skrbkdc.envar was used as a guide when making the updates.

The SKDC_TKT_ENCTYPES variable was updated to add the aes256-cts-hmac-sha1-96 and aes128-cts-hmac-sha1-96 encryption types.

```
SKDC_TKT_ENCTYPES=aes256-cts-hmac-sha1-96,aes128-cts-hmac-sha1-96,des3-cbc-sha1,
   des-hmac-sha1,des-cbc-md5,des-cbc-md4,des-cbc-crc
```

Enabling the NAS principals: In order to enable a principal for AES encryption, the password needs to be changed on a z/OS V1R9 image.

Tip: Be aware that the LISTUSER command displays the available encryption types. For example, consider the following command:

```
lu ldapeim noracf kerb
```

which returns the following results:

```
USER=LDAPEIM

KERB INFORMATION
----------------
KERBNAME= LDAP/ja0eip.pdl.pok.ibm.com
KEY VERSION= 006
KEY ENCRYPTION TYPE= DES DES3 DESD AES128 AES256
```

Notice that AES128 and AES256 are listed for the key encryption type. This is misleading! Listing of the AES128 and AES256 encryption types only indicates that they are available; it does not mean that they are active with keys associated. Again, the password must be changed in order to create the keys for the AES128 and AES256 encryption types. There is currently no way to determine whether or not the keys are available for a given encryption type. Marketing requirement MR0816074757 has been opened against RACF to address this concern.

Verifying AES encryption with Network Authentication Service

We used traces on the NAS server and on the clients to verify the operation of AES encryption.

Tracing on the server and clients: To verify that AES encryption is being used turn on tracing for both the NAS server and the client. Tracing is only needed on the server from the image where the transactions are initiated. So, for example, if

you are logged on to System A and doing a transaction to System B, only the System A server's log will show signs of encryption.

To turn on the tracing for the server, update the /etc/skrb/home/kdc/envar file. Ensure the following are active:

```
_EUV_SVC_DBG_MSG_LOGGING=1
_EUV_SVC_DBG=*.8
_EUV_SVC_DBG_FILENAME=/tmp/kerberos.%.out
```

The trace data will be written to the images /tmp directory in the kerberos.%.out file. The % is replaced with a random number.

To turn on tracing for the client, issue the following exports from the z/OS UNIX session where the commands will be issued:

```
export _EUV_SVC_DBG=*.8
export _EUV_SVC_DBG_FILENAME=krb_client.trc
export _EUV_SVC_DBG_MSG_LOGGING=1
```

The trace data will be written to the working directory in the krb_client.trc file.

Verifying the hardware: In the host system's trace file, /tmp/kerberos.%.out, there will be a section at the beginning of the trace that indicates the available hardware encryption levels.

The following output shows how the trace appeared for one of our systems. Notice that the AES128 crypto assist is available and AES256 is not currently available.

```
070810 19:06:40 (00000000) DBG1 KRB/KRB_GENERAL krb5_crypto_initialization(): Enabling strong software cryptographic support
070810 19:06:40 (00000000) DBG1 KRB/KRB_GENERAL krb5_crypto_initialization(): ICSF FMID is HCR7750
070810 19:06:40 (00000000) DBG1 KRB/KRB_GENERAL krb5_crypto_initialization(): DES crypto assist is available
070810 19:06:40 (00000000) DBG1 KRB/KRB_GENERAL krb5_crypto_initialization(): DES3 crypto assist is available
070810 19:06:40 (00000000) DBG1 KRB/KRB_GENERAL krb5_crypto_initialization(): AES128 crypto assist is available
070810 19:06:40 (00000000) DBG1 KRB/KRB_GENERAL krb5_crypto_initialization(): AES256 crypto assist is not available
```

Verifying the NAS server: In the host system's trace file, /tmp/kerberos.%.out, there will be statements indicating the enabled encryption levels. These are the default_tkt_enctypes and default_tgs_enctypes variables, updated in the krb5.conf file. Verify that both AES128 and AES256 are enabled.

The following output shows how the trace appeared for one of our systems:

```
070810 19:06:40 (00000000) DBG8 KRB/KRB_GENERAL parse_line(): Line: default_tkt_enctypes = aes256-cts-hmac-sha1-96,aes128-cts-hmac-s
ha1-96,des3-cbc-sha1,des-hmac-sha1,des-cbc-md5,des-cbc-md4,des-cbc-crc
070810 19:06:40 (00000000) DBG8 KRB/KRB_GENERAL parse_line(): Line: default_tgs_enctypes = aes256-cts-hmac-sha1-96,aes128-cts-hmac-s
ha1-96,des3-cbc-sha1,des-hmac-sha1,des-cbc-md5,des-cbc-md4,des-cbc-crc
```

Verifying the clients: We used four different clients to verify NAS's use of AES encryption: EIM, LDAP, z/OS FTP, and xLinux FTP. For information about how we enabled each of these clients for use with NAS, see our previous test reports as listed in "Network Authentication Service configuration and verification" on page 32.

Prior to issuing each client transaction, a **kinit** transaction must be issued to obtain the Kerberos credentials for the principal to be used within the client transaction. With the client tracing set, here is what you should expect to see in the trace to confirm the use of the AES encryption:

```
23317  070814 15:34:29 (00000029) DBG8 KRB/KRB_CRYPTO k5_aes_encrypt(): Software AES256 encryption performed for 16 bytes
23318  070814 15:34:29 (00000029) DBG8 KRB/KRB_CRYPTO k5_aes_decrypt(): Software AES256 decryption performed for 142 bytes
```

Notice that you should see both encryption and decryption messages.

In these examples, software was also used for the encryption and decryption. This is because our hardware is currently not enabled for AES256. A test was done using AES128 encryption. We do have a machine where the hardware is enable for AES128 encryption. Here are the type of messages you would expect to see in that case:

```
070823 11:29:00 (00000000) DBG8 KRB/KRB_CRYPTO k5_aes_encrypt(): Clear key AES128 encryption performed for 122 bytes
```

In this example the "Clear key" designation in the message is the indication that crypto hardware was used in place of software for the encryption and decryption.

Tip: If you are using Kerberos with FTP, be aware that there are two principals used within the transaction. The level of encryption used may vary depending upon the encryption types enabled for each principal. The two principals used in the transaction will be the initial principal obtained via the **kinit** command. Then the KDC uses the FTP service principal to communicate with the FTP server. When executing the initial FTP transactions, the trace files showed that two different encryption levels were being used. This was due to the FTP service principal not having its password changed within the z/OS V1R9 environment to enable the AES keys. Again, there is no current way to determine the encryption keys that are enabled for the various encryption types of a given principal. When in doubt, change the principals' password to enable the keys for the various encryption types.

KEYTAB file verification

During testing of the Network Authentication Service using FTP, we found a problem where the version of the FTP service principal listed in the KEYTAB file was out of sync with that of the FTP service principal's KERB segment in RACF.

Diagnosing the problem

While executing the FTP transaction, the following error message was displayed:

```
535-GSSAPI error major status code: d0000 - EUVF02016E Security mechanism detects error
535-GSSAPI error minor status code: 96c73ab5 - Key table entry is not found
535 Request to accept security context failed
```

We found the following information about the error minor status code, 96c73ab5, in *z/OS Integrated Security Services Network Authentication Service Administration*:

> **96C73AB5 Key table entry is not found.**
>
> **Explanation:** The requested key table entry was not found in the key table.
>
> **User response:** No action is required.

From the z/OS UNIX shell, we issued the **keytab list** command to display the contents of the keytab file. The response is similar to the following:

```
Key table: /etc/skrb/krb5.keytab

Principal: ftp@IBM.COM
  Key version: 1
  Key type: 56-bit DES
  Entry timestamp: 2007/06/12-10:55:45

Principal: ftp@IBM.COM
  Key version: 1
  Key type: 56-bit DES using key derivation
  Entry timestamp: 2007/06/12-10:55:45
```

```
Principal: ftp@IBM.COM
  Key version: 1
  Key type: 168-bit DES using key derivation
  Entry timestamp: 2007/06/12-10:55:45
```

Notice that the value of the key version is 1.

We use RACF for the KDC. To display the associated FTP principal in the RACF KDC, we issued the following LISTUSER command at the TSO Ready prompt:

```
LU FTP NORACF KERB
```

The command response is similar to the following:

```
USER=FTP

KERB INFORMATION
----------------
KERBNAME= ftp
KEY VERSION= 004
KEY ENCRYPTION TYPE= DES DES3 DESD AES128 AES256
```

Notice here that the value of the key version is 004.

So, even though the same principal exists in both the keytab file and the RACF KDC, the **ftp** command using Kerberos will fail because the key versions are not the same. What was misleading for us was that the FTP principal did exist in the keytab file. It took awhile before we realized that the key version mismatch was the problem.

Another problem that we had was that when we added the principal to the keytab file, we did not specify the key version. The **keytab add** command does not require that the key version be specified either, as it does for the password. If the key version is not specified on the **keytab add** command, it defaults to a value of 1.

Resolving the problem

To correct this condition, the principal with the correct key version needs to be added to the keytab file. We also removed the existing principal from the keytab file. Here are the series of commands we issued. These commands are all issued from the z/OS UNIX shell. Note that `ftp` is used as the principal in these examples. Replace `ftp` with the principal you will be adding to the keytab file.

1. Remove the existing principal:

   ```
   keytab delete ftp
   ```

2. List the keytab contents just to make sure the FTP principal is not there:

   ```
   keytab list
   ```

3. Add the principal into the keytab file using the key version found in the KDC. In this example, we use a key version of 004 to match the earlier example:

   ```
   keytab add ftp -p password -v 004
   ```

Additional resolution actions

We took some additional actions to help you avoid the amount of time we spent on this problem.

The first was to update informational APAR II13471 and open a publication update request for the 96C73AB5 error minor status code. Currently, *z/OS Integrated Security Services Network Authentication Service Administration* indicates that no

action is required as the user response for the error minor status code. The APAR and publication update state that the user response should be:

> List the entries in the key table file and if there is no entry for the principal used by the application then you will need to add one with the correct version number. If there is an entry already there, you will need to verify that the version number in the key table entry matches the version number for the same principle in the KDC database. If the KDC database has more than one entry for the principle, you need the highest version number.

The second action was to submit a marketing requirement. The marketing requirement number is MR0803074720. It requests a function to verify the contents of the keytab file with an associated KDC. If an out of sync condition exists, the function will flag it.

FTP Kerberos single signon support

The z/OS Communications Server FTP has been enhanced in the z/OS V1R9 release to enable FTP to use Kerberos for single signon. We'll describe our experiences with enabling z/OS Communications Server FTP for Kerberos single signon and then its exploitation.

We used the following documentation to help us with the enablement and exploitation:

- *z/OS Integrated Security Services Network Authentication Service Administration*, SC24-5926
- *z/OS Communications Server: IP Configuration Guide*, SC31-8775
- *z/OS Communications Server: New Function Summary*, GC31-8771

The client will still be prompted for a user ID during the authorization phase of the FTP transaction. What this new function prevents is the prompting and requirement of the password for the submitted user ID if it matches the ID in the Kerberos credentials that are received.

Enabling FTP Kerberos single signon support

This discussion will be just for the enablement of z/OS Communication Server FTP for single signon using Kerberos. See the discussion in our December 2005 test report for our enablement of z/OS Communication Server FTP with Kerberos.

To eliminate the client password prompt, code the following statement in the server's FTP.DATA configuration file:

```
SECURE_PASSWORD_KERBEROS OPTIONAL
```

The default value of the SECURE_PASSWORD_KERBEROS statement is `REQUIRED`. If you want to use this function, you must update the statement. In our experience, we got caught by not updating this statement. We did have to enter a password regardless of whether or not the user ID we entered matched the ID in the Kerberos credentials. So we can safely say that if you want to always have a prompt for the password, coding the statement value as `REQUIRED` will do it for you.

Verifying FTP Kerberos single signon support

A **kinit** command was first issued to obtain our Kerberos credentials. Two **ftp** commands were then issued to verify the enablement. First, when prompted for the user ID, the ID in the Kerberos credentials was entered. A password was not required for this condition, as expected. For the second **ftp** command, at the

prompt for the user ID, the ID entered did not match the Kerberos credentials. A password was required for this condition, again as expected.

The following figures show how the two transactions looked.

Figure 7 shows an FTP transaction where the user ID matches the Kerberos credentials.

```
Using /u/smith/ftp.data for local site configuration parameters.
IBM FTP CS V1R9
FTP: using TCPIP
Connecting to: host_name host_ip port: 21.
220-FTPD1 IBM FTP CS V1R9 at host_name, 19:57:26 on 2007-06-29.
220 Connection will close if idle for more than 5 minutes.
>>> AUTH GSSAPI
334 Using authentication mechanism GSSAPI
>>> ADAT
235 ADAT=YGgGCSqGSIb3EgECAgIAb1kwV6ADAgEFoQMCAQ+iSzBJoAMCAQGiQgRA
Ol4LlQI557FV1w3g7DHnE7qQiyWOgdM3KLY9fXUIRYwPDzU8U3UQxxNcoVYBQyxHv
nwGWYn6ZtjNEG/cAxzM5g==
Authentication negotiation succeeded
NAME (je0eip.pdl.pok.ibm.com:SMITH):
smith
>>> USER smith
230-User SMITH is an authorized user
230 SMITH is logged on.  Working directory is "SMITH.".
Command:
quit
>>> QUIT
221 Quit command received. Goodbye.
```

Figure 7. FTP transaction in which the user ID matches the Kerberos credentials

Figure 8 shows an FTP transaction where the user ID does not match the Kerberos credentials.

```
Using /u/smith/ftp.data for local site configuration parameters.
IBM FTP CS V1R9
FTP: using TCPIP
Connecting to: host_name host_ip port: 21.
220-FTPD1 IBM FTP CS V1R9 at host_name, 20:03:04 on 2007-06-29.
220 Connection will close if idle for more than 5 minutes.
>>> AUTH GSSAPI
334 Using authentication mechanism GSSAPI
>>> ADAT
235 ADAT=YGgGCSqGSIb3EgECAgIAb1kwV6ADAgEFoQMCAQ+iSzBJoAMCAQGiQgRA
KKrEYfNHzX3dyHlf1AH1FZDUx4mJ4On/rD1W8hKPilU5DDkVPNYgiBs7iVJjxLm76
XMF6ZspcdjCDAgu/ZtgoQ==
Authentication negotiation succeeded
NAME (je0eip.pdl.pok.ibm.com:SMITH):
jones
>>> USER jones
331 Send password please.
PASSWORD:

>>> PASS
230 JONES is logged on.  Working directory is "JONES.".
Command:
quit
>>> QUIT
221 Quit command received. Goodbye.
```

Figure 8. FTP transaction in which the user ID does not match the Kerberos credentials

Using LDAP Server

The LDAP Server is a component of z/OS Security Server which uses the Lightweight Directory Access Protocol (LDAP) standard, an open industry protocol for accessing information in a directory. For V1R8, there are two versions of the LDAP server available:

1. Integrated Security Services (ISS) Server
2. IBM Tivoli Directory Server (TDS)

We address the following topics related to using LDAP Server:

- "Overview of our LDAP configuration"
- "Using AES encryption with IBM Tivoli Directory Server" on page 44

Overview of our LDAP configuration

We have a multiplatform LDAP configuration for both the Integrated Security Services (ISS) LDAP environment and the IBM Tivoli Directory Server (IBM TDS) environment. The following figures illustrate both environments followed by a listing of exploiters of each environment.

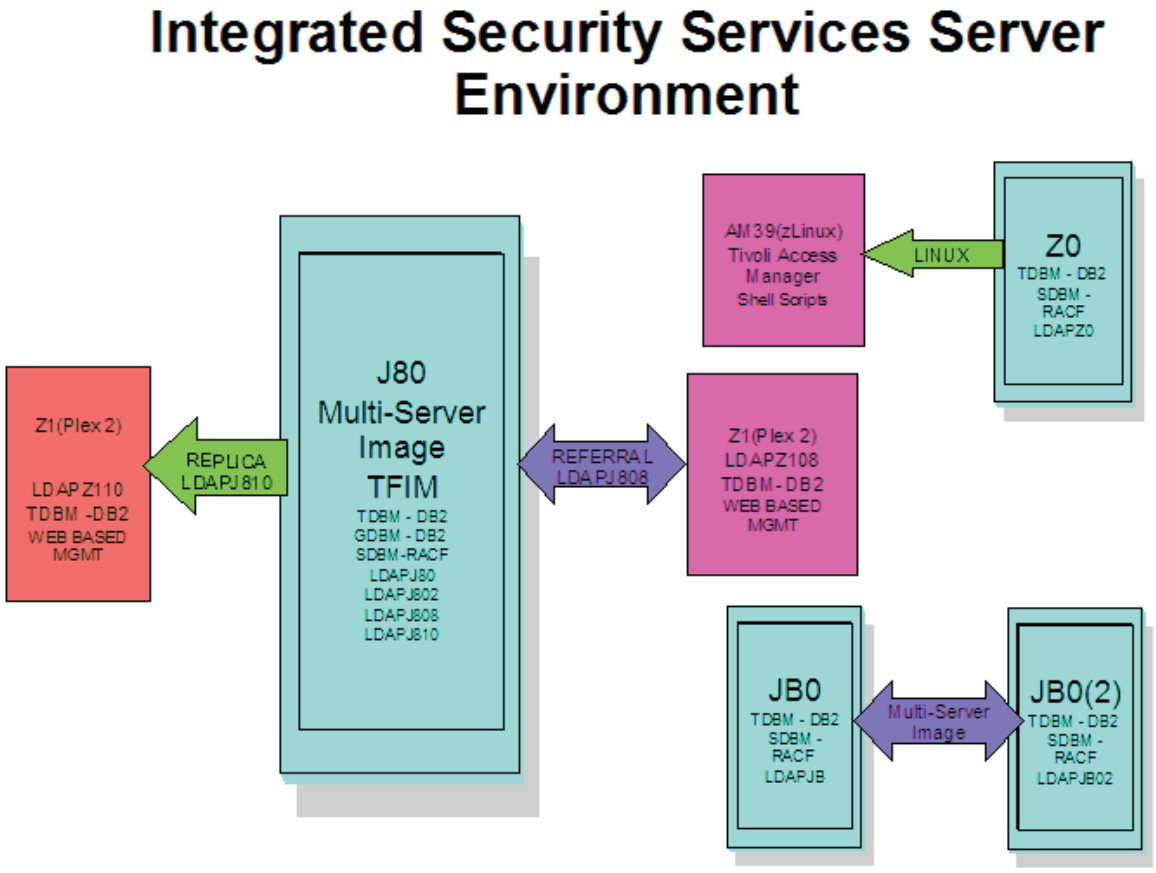


Figure 9. Integrated Security Services (ISS) LDAP environment

DB2 has a TDBM backend, connecting LDAP to the DB2 Database Directory. It also has a GDBM backend. The GDBM backend is used to store Change

Log entries created as a result of RACF modifications. These environments are exploited using scripts run from Windows agents that allow stress to be placed on the z/OS LDAP Servers using DB2.

RACF has a SDBM backend, connecting to the RACF directory found on our plex.

Integrated Security Services (ISS) LDAP exploitation

LDAP Referral

This configuration is set up between a LDAP Server on Plex 2 (LDAPZ108) and a LDAP Server on Plex 1 (LDAPJ808) both using TDBM backends. The Plex 2 LDAP Server (LDAPZ108) has a general referral found in its configuration file that points to a master LDAP server on z/OS (LDAPJ808). This allows the user to run the **ldapsearch** command from the LDAP server on Plex 2 for an entry that is not found in that directory, but may be found in the LDAPJ808 master server's directory. The command will return all entries found that match from both directories.

Replication (Master/Slave)

We run our ISS replication transactions between LDAPJ810 and LDAPZ110. Replication functions quite like the Stress operation above but with one important difference. The master receives the new entry and its modifications and eventual deletion. However the slave, which has been initialized like the master, is checked twice, the first time after the entry is added to insure it has been replicated on the slave and also after the deletion to insure it has indeed been deleted from the slave through the replication process. The checking process is repeated until it is either found (during the add) or not found (during the deletion) or the server reaches a specified search count (which causes a failure).

Replication (Peer/Peer)

We run our ISS replication transactions between LDAPJ810 and LDAPZ110. Peer to peer replication functions similarly to master/slave except that each server takes turns at being the ″master″, that is having its entries manipulated by the program while the other server is checked for entry availability. When the program is run in a loop, the ″master″ and ″slave″ switch places on each new loop cycle.

Persistent Search

We run our ISS persistent search transactions between LDAPJ810 and LDAPZ110. The persistent search function detects the revisions that have been made to a server's entries and prints out the results, the detail depending on the display level setting. The program is initiated with the entry filter and operation monitor parameters set and it will listen to the designated server until a specified entry type operation is encountered for reporting. This repeats until the program is terminated. Of course for this function to operate, there must be some activity on the server being monitored. That is one use for the Stress function. An instance of it can be run to stimulate the desired server. Also, the persistent search could be directed against one of the replication servers if desired. For another workload scenario, several instances of the persistent search can be run, with each detecting a different change type (or combination thereof).

Tivoli Access Manager on zLinux

We have set up Tivoli Access Manager (TAM) on our zLinux SUSE 8 machine to enable cross platform testing between Linux and z/OS. TAM uses z/OS LDAP as a backend to store userid information that will either allow or deny user access to TAM. Testing is done using Shell scripts run on Linux that allow stress to be placed on the z/OS LDAP Server on Z0.

Tivoli Federated Identity Manager on zLinux
We have setup Tivoli Federated Identity Manager (TFIM) on our zLinux machine to enable cross platform testing between Linux and z/OS. TFIM uses z/OS LDAP as a backend to store userid information in a similar capacity to TAM. However, our TFIM setup requires the use of two LDAP Servers; LDAPJ80 and LDAPJ802. This environment is exploited using Shell scripts run from Windows agents that allow stress to be placed on the z/OS LDAP Servers on J80.

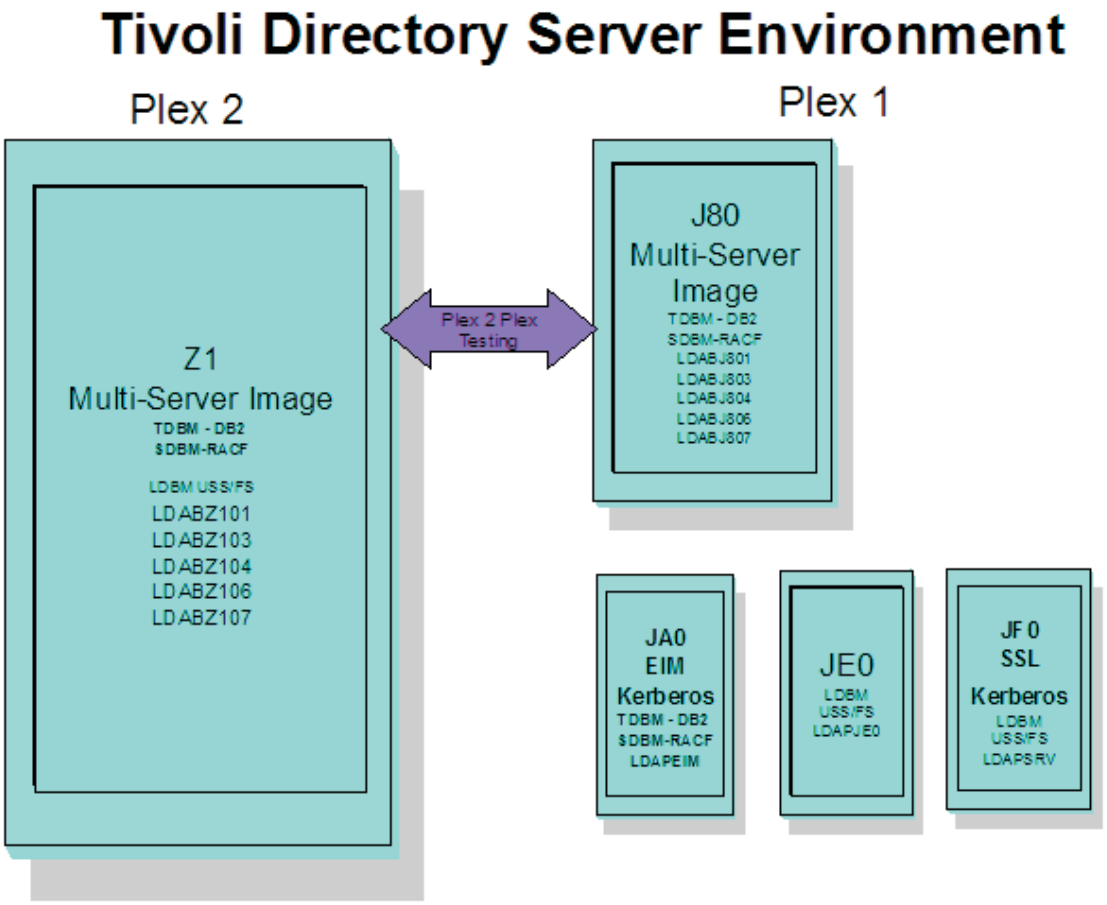


Figure 10. IBM Tivoli Directory Server (IBM TDS) environment

DB2 has a TDBM backend, connecting LDAP to the DB2 Database Directory. This environment is exploited using scripts run from Windows agents that allow stress to be placed on the z/OS LDAP Servers using DB2.

RACF has a SDBM backend, connecting to the RACF directory found on our plex.

Unix System Services file system
has an LDBM backend, connecting to a Unix System Services file system on our plex. This environment is exploited in two ways. The first is with tso http servers. The IBM HTTP Server powered by Domino running one of our z/OS images and Apache running on an xLinux box. Both of these http servers access the LDAPJE0 IBM TDS for authentication to access http

resources. The second is to drive Kerberos transactions using shell scripts run from within our USS environment. This workload accesses the LDAPJF0 IBM TDS.

IBM Tivoli Directory Server (IBM TDS) exploitation

Kerberos
We currently have two LDAP servers on our plex that are setup for Kerberos transactions. They are LDAPSRV on JF0 and LDAPEIM on JA0.

EIM We currently have one LDAP server on our plex that is setup for EIM transactions. It is LDAPEIM on JA0.

LDAP Referral
This configuration is set up between a LDAP Servers on Plex 2 and a LDAP Servers on Plex 1 (LDABJ804/LDABJ806) using the TDBM and LDBM backends. The Plex 2 LDAP Servers (LDABZ104/LDABZ106) have a general referral found in its configuration file that points to master LDAP servers on z/OS (LDABJ804/LDABJ806). This allows the user to run the **ldapsearch** command from the LDAP servers on Plex 2 for an entry that is not found in that directory, but may be found in the master servers' directories. The command will return all entries found that match from both directories.

Replication (Master/Slave)
We run our IBM TDS master/slave replication transactions between LDABJ801 and LDABZ101. Replication functions quite like the Stress operation above but with one important difference. The master receives the new entry and its modifications and eventual deletion, but the slave, which has been initialized like the master, is checked twice, the first time after the entry is added to insure it has been replicated on the slave and also after the deletion to insure it has indeed been deleted from the slave through the replication process. The checking process is repeated until it is either found (during the add) or not found (during the deletion) or the server reaches a specified search count (which causes a failure).

Replication (Peer/Peer)
We run our IBM TDS peer/peer replication transactions between LDABJ803 and LDABZ103. Peer to peer replication functions similarly to master/slave except that each server takes turns at being the ″master″, that is having its entries manipulated by the program while the other server is checked for entry availability. When the program is run in a loop, the ″master″ and ″slave″ switch places on each new loop cycle.

Persistent Search
We run our IBM TDS persistent search transactions between LDABJ804 and LDABZ104. The persistent search function detects the revisions that have been made to a servers entries and prints out the results, the detail depending on the display level setting. The program is initiated with the entry filter and operation monitor parameters set and it will listen to the designated server until a specified entry type operation is encountered for reporting. This repeats until the program is killed. Of course for this function to operate there must be some activity on the server being monitored. That is one use for the Stress function. An instance of it can be run to stimulate the desired server. Also, the persistent search could be directed against one of the replication servers if desired. For another workload scenario, several instances of the persistent search can be run, with each detecting a different change type (or combination thereof).

Using AES encryption with IBM Tivoli Directory Server

IBM Tivoli Directory Server (IBM TDS) was enhanced in the z/OS V1R8 release to provide AES encryption. We'll talk about our experiences with enabling AES encryption and then the exploitation of it.

We used the information in *IBM Tivoli Directory Server Administration and Use for z/OS*, SC23-5191 to help us with the enablement and exploitation.

Enabling AES encryption with IBM Tivoli Directory Server

Although AES encryption was made available with the initial release of IBM TDS during the z/OS V1R8 time period, we are now reporting on its enablement in this our z/OS V1R9 test report.

As a word of caution, the Integrated Security Services (ISS) LDAP Server is not enabled for AES encryption. Special consideration should be taken if the IBM TDS server will be interacting with an ISS LDAP Server regarding the use of AES encryption. See *IBM Tivoli Directory Server Administration and Use for z/OS* for details.

The first step—and the one that was the most confusing for us—was to create the data set that would hold the AES key. It turns out that this is just a simple sequential data set and the ISPF editor is used to enter the key label and key parts. The documentation does explain this but we thought that there would be some kind of tool to create and maintain these keys. We didn't think it would be this simple but it is. We created a sequential data set, IBMTDS.LDAP.AESKEYS, and added the following for the AES key:

```
LDAPSRV  123456789ABCDEF0  23456789ABCDEF01  ABCDEF0123456789  F9E8D7C6B5A43210
```

The format for the key is:

```
key-label key-part-1 key-part-2 key-part-3 key-part-4
```

The next step is to add the LDAPKEYS DD statement to the servers startup procedure to point to the AES key data set that we just created:

```
//LDAPKEYS DD DSN=IBMTDS.LDAP.AESKEYS,DISP=SHR
```

The userPassword attribute values were used for the AES encryption verification. To enable this, we updated the configuration file with the following:

```
pwEncryption AES:LDAPSRV
```

Notice that LDAPSRV matches the key-label of the record in the IBMTDS.LDAP.KEYS data set for the AES key to use.

The secretKey or replicaCredentials attributes are also available for AES encryption. These attributes have not yet been enabled in our environment.

IBM TDS was then recycled to pick up the changes to the configuration file and the startup procedure.

Verifying AES encryption with IBM Tivoli Directory Server

After IBM TDS was recycled, the server's JES log contained the following statements which helped validate the AES encryption enablement:

- In the DSOUT:

  ```
  pwEncryption: AES:LDAPSRV
  ```

- In the JESMSGLG:

```
IAT4401  LOCATE FOR STEP=GO       DD=LDAPKEYS DSN=IBMTDS.LDAP.AESKEYS
```

To AES encrypt the user password in the IBM TDS backend, the password must be changed. To do this, we used the **ldapmodify** command along with an ldif file which contained the password change. To verify that the actual encryption takes place, we turned on tracing using this console command:

```
MODIFY ldap_started_task,DEBUG ALL
```

Here is an example of the ldif file contents:

```
dn: cn=Eddie Catu, ou=In Flight Systems, ou=Home Town, o=Your Company
changetype: modify
replace:x
userpassword: catu
```

Notice the userpassword is set to change to `catu`.

Then, we issued the following **ldapmodify** command to make the change:

```
ldapmodify -h host_ip -D "cn=LDAP Administrator, o=Your Company" -w admin_password
  -f /directory/EddieCatu.ldif
```

We found the following in the trace after the **ldapmodify** command:

```
*c..f}.Acn=Eddie *
*Catu, ou=In Flig*
*ht Systems, ou=H*
*ome Town, o=Your*
* Company0806...0*
*1..userpassword1*
*!0...{AES:......*
*.}d...|z.....LTf*
*%...".%cn=LDAP A*
*dministrator, o=*
*Your Company..20*
*070731225431.432*
*914Z.........I  *
```

Notice the string `{AES:` which indicates the beginning of the AES encryption. The closing brace (}) indicates the end of the encryption.

We issued the following console command to turn off tracing:

```
MODIFY ldap_started_task,DEBUG 0
```

As a final verification of the AES encryption, the admin ID was used to display the Eddie Catu user password entry in the IBM TDS backend. Here is an example of the **ldapsearch** command used to display that data:

```
ldapsearch -h host_ip -D "cn=LDAP Administrator, o=Your Company" -w admin_password
  -b "cn=Eddie Catu,ou=In Flight Systems,ou=Home Town,o=Your Company"  "(objectclass=*)" userpassword
```

The following results were returned:

```
cn=Eddie Catu,ou=In Flight Systems,ou=Home Town,o=Your Company
userpassword=catu
```

Notice that the userpassword returned was `catu`.

Initially, we expected the userpassword to be returned encrypted. However, *IBM Tivoli Directory Server Administration and Use for z/OS* does indicate that AES encryption is two-way encryption. This means that the data is stored encrypted and it is returned from a query unencrypted.

Cryptographic Services PKI Services updates

For z/OS V1R9, there were a few updates for PKI Services that we'd like to highlight.

Automatic certificate renewal

This new PKI ability allows the user to set up certificates for automatic renewal, along with email notification of certificate renewal.

We did the following to enable automatic certificate renewal:

1. Set the ExpireWarningTime variable in the CertPolicy section of pkiserv.config:

   ```
   EXPIREWARNINGTIME = 1W
   ```

2. Copied and customized the renewed certificate notification form. See *z/OS Cryptographic Services PKI Services Guide and Reference* for detailed information on this step.
3. Updated the General Section of pkiserv.config to point to the newly configured form:

   ```
   RENEWCERTFORM=Z0/ETC/PKISERV/RENEWCERTFORM.FORM
   ```

4. We went back to the certificate template that we want to renew and enabled automatic renewal by adding the <AUTORENEW=Y> tag to the template. It is recommended to place this tag after the <NICKNAME> tag in the template:

   ```
   <NICKNAME=1YBSSL>
   <AUTORENEW=Y>
   ```

5. Enabled e-mail notification in the certificate template by changing the original setting for the NotifyEmail tag from:

   ```
   %%NOTIFYEMAIL (OPTIONAL)%%
   ```

   to:

   ```
   %%NOTIFYEMAIL%%
   ```

Automatic certificate renewal was successfully enabled.

RACF/SDBM distinguished name support

PKI Services now allows the configuration of RACF-style distinguished names with IBM Tivoli Directory Server. To verify this we configured PKI Services with the following changes:

From the old distinguished name:

```
CN=LDAP ADMINISTRATOR
```

To the new distinguished name:

```
RACFID=WEBADM, PROFILETYPE=SYSPLEX=UTCPLXJ8,C=US
```

The password field was also updated.

The configuration change was successful and PKI functioned as normal.

Using System SSL

We have one new System SSL enhancement to report on for z/OS V1R9.

System SSL hardware to software notification

For z/OS V1R9, System SSL has been enhanced to provide information about when an application has switched from using hardware encryption processing to software encryption processing. If a System SSL application encounters an error when using hardware services through ICSF, System SSL automatically switches to software services for that encryption work. In z/OS V1R9, System SSL added notification via messages to the console and system log that this switch has taken place.

For example, when we stopped ICSF while attempting an SSL connection through the IBM HTTP Server, the SSL connection completed successfully but when looking in the log, we saw the following messages:

```
GSK01051E IMWEBZ1/01FD Hardware encryption error. ICSF hardware encryption processing is unavailable.
GSK01052W IMWEBZ1/01FD Hardware encryption error. PKE encryption processing switched to software.
```

Implementing and using PKCS #11 support

In z/OS V1R9, ICSF provides support for the PKCS #11 standard. PKCS #11 is the standard for the cryptographic token interface. z/OS' implementation of the PKCS #11 tokens are virtual, very similar to RACF (SAF) key rings. z/OS PKCS #11 tokens can be created using system software, such as the gskkyman utility, RACF, or by applications using the C API. ICSF supports PKCS #11 tokens as follows:

- A token data set called a TKDS that serves as the repository for cryptographic keys and certificates used by PKCS #11 applications
- A C application programming interface (API) that supports a subset of the V2.20 level of the PKCS #11 specification
- Token management callable services that are used by the C API

Setting up PKCS #11 supuport

For setup of PKCS #11 support, we used *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*, SA23-2231. Using this documentation, we performed the following steps:

1. Created TKDS VSAM data set, SYS1.TKDS1, using the sample provided in SYS1.SAMPLIB(CSFTKDS) as the base.
2. Updated the ICSF options dataset to specify the necessary TKDS options. We specified the following to indicate the name of our TKDS data set:

   ```
   TKDSN(SYS1.TKDS1)
   ```

   And the following to indicate that we wanted to share our TKDS across all systems in our sysplex:

   ```
   SYSPLEXTKDS(YES,FAIL(YES))
   ```
3. The CRYPTOZ class is used to control access to the tokens. We chose to setup generic profiles so we activated the CRYPTOZ class as follows:

   ```
   SETROPTS CLASSACT(CRYPTOZ) GENERIC(CRYPTOZ) RACLIST(CRYPTOZ)
   ```
4. We created the USER.* and SO.* resources and gave CONTROL authority for both to the ID creating the tokens:

   ```
   RDEFINE CRYPTOZ SO.* UACC(NONE)
   RDEFINE CRYPTOZ USER.* UACC(NONE)
   PERMIT SO.* CLASS(CRYPTOZ) ID(TOKADMIN) ACC(CONTROL)
   PERMIT USER.* CLASS(CRYPTOZ) ID(TOKADMIN) ACC(CONTROL)
   SETROPTS RACLIST(CRYPTOZ) REFRESH
   ```

   Note that any ID whose applications attempt to access tokens will need READ access to the appropriate resources (SO and USER).

5. We tested to ensure our setup was correct by running the pre-compiled version of testpkcs11.

Using the PKCS #11 support

We tested the PKCS#11 support by using the available applications that support PKCS #11 tokens.

Using gskkyman and the new panels for tokens, we created a token the same way we would create a certificate/key. This was fairly straight forward. For more information about how to create tokens using the gskkyman interface, see *z/OS Cryptographic Service System Secure Sockets Layer Programming*.

We used this token in 2 ways:

- Created an SSL connection on the HTTP servers. In the config file for the HTTP server, we indicated the following for the KeyFile directive:

  ```
  KeyFile *TOKEN*/token.name SAF
  ```

  where *token.name* is the actual name given to the token.
- Created an SSL connection while using FTP. Much like the HTTP server above, we defined the Keyring directive in the FTP config file as follows:

  ```
  Keyring *TOKEN*/token.name SAF
  ```

  where *token.name* is the label given to token upon creation.

You need to ensure that the IDs that the FTP server and HTTP server are running under have READ access to the profiles (USER.* and SO.*) in the CRYPTOZ class; otherwise, things will not work. For example, when starting the HTTP server without the proper authority, the HTTP server will start but you will see the following messages in the log:

```
ICH408I USER(WEBSRV  ) GROUP(IMWEB   ) NAME(####################)  791
  USER.IMWEBZ1.CERTS.JUNE12 CL(CRYPTOZ )
  INSUFFICIENT ACCESS AUTHORITY
  FROM USER.* (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IMW6310E SSL support initialization failed, server will run only in non-secure mode
ICH408I USER(WEBSRV  ) GROUP(IMWEB   ) NAME(####################)  792
  SO.IMWEBZ1.CERTS.JUNE12 CL(CRYPTOZ )
  INSUFFICIENT ACCESS AUTHORITY
  FROM SO.* (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
```

We also tested that the C API interface worked correctly, using a home grown workload.

Implementing and using the RACF Java API

z/OS V1R9 introduces a Java API that includes some basic administrative functions that can be used with RACF. This API does not include all RACF functionality nor does it add any new RACF functionality—it takes advantage of what is already possible with LDAP SDBM, an LDAP server with a RACF backend.

In order to test this API, we first needed to implement the infrastructure and then we created a J2EE Web application that would let us manage our RACF users and groups. For simplicity, we called our Web application zRacfAdmin. Figure 11 on page 49

page 49 shows our setup.

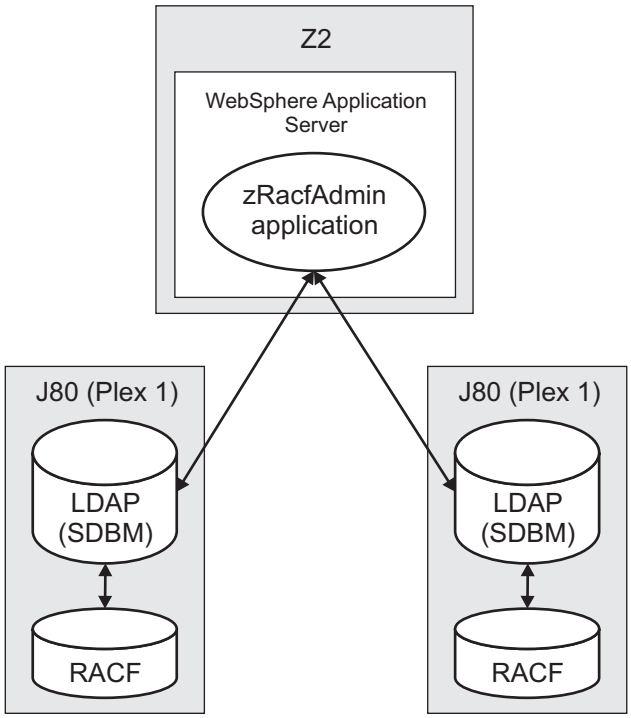


Figure 11. Our setup for testing the RACF Java API

As Figure 11 shows, we were able to manage two different RACF databases in separate sysplexes by writing one J2EE Web application that was able to connect to both.

The type of functionality supported by the Java API includes the following:

- Creating a connection to the LDAP server
- Creating/Editing/Deleting users and user attributes
- Creating/Editing/Deleting groups and group attributes
- Adding/Editing/Removing user-to-group connection attributes

In order to exploit the Java API, you need two jar files, both of which are located in /usr/include/java_classes. The two jar files are RACFuserregistry.jar and userregistry.jar.

The Java API is simple to use for the most part. For example, to retrieve a user object from the RACF database through LDAP, you issue a **getUser()** call. You can then use other function calls to retrieve the attributes associated with the user object.

Figure 12 on page 50 shows a screen capture from our zRacfAdmin Web application. In this figure, we are viewing the attributes of the WASADM user. We can edit any of those attributes, as well as add the user to or remove the user from various groups.

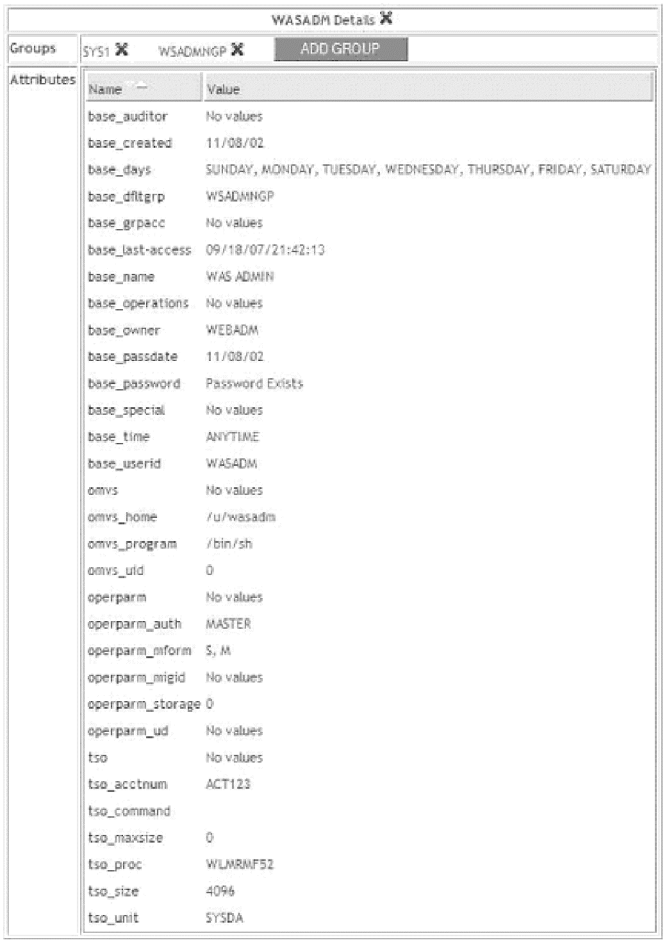


Figure 12. Our zRacfAdmin Web application, attribute display for user WASADM

The Java API does not include any searching functionality for users or groups. However, thanks to Anne Emerick (a colleague in z/OS RACF Development), we were able to use a Java-LDAP API to implement the search functionality into our Web application.

Overall, our experience with the Java API for RACF was a positive one and we also found the API useful for managing RACF databases from homegrown Web applications.

Chapter 4. Migrating to and using z/OS V1R9

This topic describes our migration to z/OS V1R9. Our migration experiences include:

- "z/OS V1R9 base migration experiences"
- "Other z/OS V1R9 migration experiences" on page 53
- "Coupling facility maintenance enhancements" on page 57
- "Testing greater than 32 CPU support" on page 58

Here we primarily discuss our sysplex-oriented migration experiences and other related experiences. This includes the enablement of significant new functions and, if applicable, performance aspects. Detailed test experiences with major new functions beyond migration and experiences with other z/OS products appear in subsequent chapters.

You can read about our migration experiences with earlier releases of z/OS in previous editions of our test report, available on our Web site.

For migration experiences with...	See this edition of our test report...
z/OS V1R8 and z/OS.e V1R8	*zSeries® Platform Test Report for z/OS and Linux Virtual Servers, June 2007*
z/OS V1R7 and z/OS.e V1R7	*zSeries Platform Test Report for z/OS and Linux Virtual Servers, December 2005*

z/OS V1R9 base migration experiences

This topic describes our experiences with our base migration to z/OS V1R9, without having implemented any new functions. It includes our high level migration process along with other migration activities and considerations.

Our high-level base migration process

The following is an overview of our z/OS V1R9 migration process.

Before we began: We reviewed the migration information in *z/OS Planning for Installation*, GA22-7504 and *z/OS Migration*.

Table 9 shows the high-level process we followed to migrate the members of our sysplex from z/OS V1R8 to z/OS V1R9.

Table 9. Our high-level migration process for z/OS V1R9

Stage	Description
Updating PARMLIB for z/OS V1R9	We created SYS1.PETR19.PARMLIB to contain all the PARMLIB members that changed for z/OS V1R9 and we used our LOAD*xx* member for migrating our systems one at a time. (See our December 1997 test report for an example of how we use LOAD*xx* to migrate individual systems.)

Table 9. Our high-level migration process for z/OS V1R9 (continued)

Stage	Description
Applying coexistence service	We applied the necessary coexistence service (also known as compatibility or toleration PTFs) to position our systems for the migration. See the coexistence service requirements in *z/OS Planning for Installation* and make sure you install the fixes for any APARs that relate to your configuration before you migrate.
IPLing our first z/OS V1R9 image	We brought up z/OS V1R9 on our Z3 test system and ran it there for a couple of weeks.
Updating the RACF® templates	To test the RACF dynamic template enhancement, we IPLed the first z/OS V1R9 image without first running the IRRMIN00 utility with PARM=UPDATE. As expected, the following message appeared: `ICH579E RACF TEMPLATES ON DATABASE ARE DOWNLEVEL` RACF initialization still completed successfully. We then ran IRRMIN00 with PARM=UPDATE to dynamically update the templates on all six RACF data sets without the need for an IPL. (See *z/OS Security Server RACF System Programmer's Guide*, SA22-7681 for details about RACF templates.)
IPLing additional z/OS V1R9 images	We continued to bring up additional z/OS V1R9 images across our sysplex, as follows: 1. Brought up z/OS V1R9 on our Z1 test system and ran with it for a week. 2. Migrated our last test system, Z2, and ran for a week. 3. Migrated some of our production systems, JA0, JE0, JC0 and J80, and ran with it for a couple of days. 4. At this point, we took two of our production V1R9 images, JC0 and JE0, back down to V1R8. This is part of our focus on migration testing and fallback. We ran for two full days and experienced no fallback issues. 5. Migrated three additional production systems, Z0, JB0, and TPN, and ran for a week. 6. Migrated the remaining production systems, JF0 and J90, to V1R9.

More about our base migration activities

This topic highlights additional details about some of the base migration activities that we perform with each new release, including running with mixed product levels, using concatenated PARMLIB, and recompiling automation EXECs.

Running with mixed product levels

During our migration, we successfully ran our sysplex with mixed product levels, including the following:

- z/OS V1R8 and z/OS V1R9
- z/OS V1R8 JES2 and z/OS V1R9 JES2
- z/OS V1R8 JES3 and z/OS V1R9 JES3

Using concatenated PARMLIB

We continue to use concatenated PARMLIB support to add or update PARMLIB members for z/OS V1R9. See our Web site for examples of some of our PARMLIB members.

This is a good use of concatenated PARMLIB because it isolates all of the PARMLIB changes for z/OS V1R9 in one place and makes it easier to migrate multiple systems. Rather than change many PARMLIB members each time we migrate another system to V1R9, we just add the PARMLIB statements at the appropriate places in SYS0.IPLPARM(LOAD*xx*) to allow that system to use SYS1.PETR19.PARMLIB.

Recompiling REXX EXECs for automation

We recompiled our IBM Tivoli® System Automation for z/OS REXX™ EXECs when we migrated to z/OS V1R9. We discuss the need to recompile these REXX EXECs in our our December 1997 test report.

Other z/OS V1R9 migration experiences

This topic highlights additional details about some of our migration experiences that are specific to z/OS V1R9.

z/OS V1R9 Unicode support enhancements

In z/OS V1R9, the LE C Run-Time Library **iconv()** family of functions is updated to use the Unicode Conversion Services. This change is generally transparent through the magic of z/OS!

One item in this area that we would like to point out, especially for those migrating from releases prior to z/OS V1R7, is that you may be able to remove any customized conversion image. In our case, we had created a customized image for DB2 support many years ago. The default conversion image that comes with z/OS V1R7 and above provides the most commonly used code page conversion tables needed to satisfy the DB2 code page conversion needs and eliminates the need to create or use a customized conversion image.

For further details, see the section “Remove CUNUNIxx parmlib members” in *z/OS Migration*.

System symbols documentation issue

During our migration to z/OS V1R9, we discovered an error in the V1R9 documentation regarding system symbols.

Starting in z/OS V1R9, the same ability to substring system symbols as described in *z/OS MVS Initialization and Tuning Reference* is now available for system symbols that are being used in JCL for started task procedures and TSO logon procedures. Previously, the ability to substring system symbols in JCL was not provided.

We discovered a problem when one of our started task procedures that offloads LOGREC data to a GDG based data set started failing. The following is an example of the procedure that failed:

```
//DUMPEREP PROC OUT=C,SYS=UNK,CAT=ABEND013,LOGDSN=ABEND013
//*************************************************************
//*  COPIES ALL THE RECORDS TO                                *
//*  'ICRWRTR.EREPXXX.G0000V00', WHERE XXX IS THE SYSTEM SMFID. *
//*************************************************************
```

```
//IEFPROC EXEC PGM=IFCEREP1,REGION=2M,
//            PARM='ACC=Y,ZERO=Y,PRINT=NO'
//OUTPUT1  OUTPUT FORMDEF=BJGR
//SERLOG    DD DSN=&LOGDSN,DISP=SHR
//EREPPT    DD SYSOUT=&OUT,OUTPUT=*.OUTPUT1,DCB=BLKSIZE=133
//TOURIST   DD SYSOUT=&OUT,OUTPUT=*.OUTPUT1,DCB=BLKSIZE=133
//DIRECTWK  DD UNIT=SYSDA,SPACE=(CYL,(10))
//ACCDEV    DD DSN=ICRWRTR.EREP&SYSNAME(0),DISP=MOD,
//             DCB=(ICRWRTR.EREP.MODEL.DSCB)
//ACCIN     DD DUMMY,DCB=BLKSIZE=133
//SYSIN     DD DUMMY,DCB=BLKSIZE=133,
//             VOL=SER=D83I80,UNIT=3390,DISP=OLD,DSN=&CAT&SYS
```

For instance, in this JCL, when the value of the &SYSNAME system symbol is Z1, the use of the &SYSNAME system symbol followed by "(0)" in the data set name ICRWRTR.EREP&SYSNAME(0) has always worked by generating a substitution value of ICRWRTR.EREPZ1(0). However, after migrating to z/OS V1R9, the substitution value that was generated was ICRWRTR.EREPZ, which caused incorrect results. The correction was to change the JCL to specify the data set name as either ICRWRTR.EREP&SYSNAME.(0) or ICRWRTR.EREP&SYSNAME(+0) (apparently, the use of the + character in substrings is not allowed).

The information in *z/OS MVS JCL Reference* will be changed to document the ability to substring system symbols in started task procedures and TSO logon procedures.

Using the IBM Migration Checker for z/OS

From the Migration Checker documentation:

> The IBM Migration Checker for z/OS is a tool composed of several batch programs that check the applicability of certain migration actions on your currently running system. You can run each batch program independently (using separate jobs) or you can run them all "at once" (serially, using a single job). The IBM Migration Checker for z/OS was introduced for migrations from z/OS V1R7 to z/OS V1R8. However, other migration paths at the time were tolerated. The tool has been subsequently enhanced for migrations to z/OS V1R9. The program detects the z/OS release, and the output indicates whether any migration information can be provided for the release.

When we begin to evaluate a new release of z/OS, the Migration Checker is a useful tool for evaluation of base components and a verification of several basic configuration options.

The latest release of the IBM Migration Checker for z/OS is available from the z/OS downloads page at www.ibm.com/servers/eserver/zseries/zos/downloads/. Do the following to obtain the Migration Checker:

1. From the z/OS downloads page, download the three binary files, which are in TSO XMIT format, to your workstation.
2. Use FTP to transfer the binary files from your workstation to your z/OS host. The following example shows the FTP commands to do this:

```
FTP JC0EIP
<enter your user ID and password when prompted>
BIN
QUOTE SITE RECFM=FB LRECL=80 BLKSIZE=32720 TRACKS PRIM=100 SEC=100
```

```
PUT migrate.checker.clist.bin
PUT migrate.checker.jcl.bin
PUT migrate.checker.load.bin
QUIT
```

3. In a TSO session issue the RECEIVE command to receive each file into a PDS. Instructions are in the documentation provided with the application.

We wrote a REXX EXEC named MC that will take the output produced by the Migration Checker and write a sequential data set of HTML code that can be viewed with a Web browser. The REXX EXEC is invoked using the following JCL:

```
//AJNIMSS1 JOB 'SDSFTST','AL NIMS',REGION=4M,
//  CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=AJNIMS
//*
//MC       EXEC PGM=IKJEFT01,PARM='%MC J80'
//SYSPROC  DD DISP=SHR,DSN=AJNIMS.CLIST
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//MIGIN    DD  DISP=SHR,DSN=AJNIMS.ALPHA.MIGRATE.CHECKER.J80.OUTPUT
//HTMLOUT  DD  DISP=OLD,DSN=AJNIMS.MIGRATE.CHECK.HTML(J80)
//*
//*
```

The data set identified by the MIGIN ddname contains the output produced by the Migration Checker. The data set identified by the HTMLOUT ddname will contain the HTML formatted output. The output consists of sequential data, so it can be placed either in a PS or PO data set.

Our MC EXEC uses IBM SmartBatch for OS/390 BatchPipeWorks. The following is the REXX code for our MC EXEC:

```
/*** REXX *** */

Arg MCSys

Address "TSO"

If MCSys = '' Then Do
  Say "No System Specified"
  Exit
End

/*********************************************************************/
/*                                                                   */
/* DDNames Used: MIGIN <- Output Dataset from the IBM Migration      */
/*                        Checker for z/OS Run.                      */
/*               HTMLOUT <- HTML Output File Destination.            */
/*                          Output is a single Sequental             */
/*                          Data Set Output.                         */
/*                                                                   */
/*********************************************************************/

DDInfo = LISTDSI('MIGIN' 'FILE')

If DDInfo > 4  Then Do
  Say '-----------------------------------------------------------'
  Say 'DDNAME: MIGIN was not allocated.'
  Say 'DDNAME(MIGIN) Should be allocated to the'
  Say ' IBM Migration Checker for z/OS Output Dataset.'
  Say '-----------------------------------------------------------'
  Exit 20
End
```

```
|
|                     If SYSDSORG ^= 'PO' Then Do
|                       Say '-------------------------------------------------------'
|                       Say 'The Dataset' SYSDSNAME
|                       Say 'Is not a Partitioned Dataset and not proper input for this'
|                       Say 'program.  DSORG Found:' SYSDSORG
|                       Say '-------------------------------------------------------'
|                       Exit 20
|                     End
|
|                     "PIPE LISTISPF '"||SYSDSNAME||"'",
|                         "| CHOP 8",
|                         "| STRIP BOTH",
|                         "| STEM MCMin."
|
|                     If MCMin.1 ^= '$MIGALL' Then Do
|                       Say '-------------------------------------------------------'
|                       Say 'The Dataset' SYSDSNAME
|                       Say 'The First Member in the Data Set is not $MIGALL'
|                       Say '-------------------------------------------------------'
|                       Exit 20
|                     End
|
|                     /*****************************************************************/
|                     /*                                                               */
|                     /* Load in the $MIGALL member, the INDEX Member.                 */
|                     /*                                                               */
|                     /*****************************************************************/
|                     "PIPE MEMBERS '"||SYSDSNAME||"' $MIGALL",
|                         "| STEM MigAll."
|
|                     Queue '<HTML><HEAD> '
|                     Queue '<title>Migrate Check of '||MCSys||'</title>'
|                     Queue '<STYLE type="text/css"> '
|                     Queue ' body '
|                     Queue ' { Background-color: White ; '
|                     Queue '   Color: Blue } '
|                     Queue '</STYLE></HEAD> '
|                     Queue '<BODY>'
|                     Queue ,
|                         '<center><h1>IBM Migration Checker for z/OS<br>',
|                              'System Checked:' MCSys,
|                              '</h1></center>'
|                     Queue '<P><A Name="TOP"></A></P> '
|
|                     Queue 'Index'
|                     Queue '<br><PRE>'
|
|                     /*****************************************************************/
|                     /*                                                               */
|                     /* Go through the INDEX creating "Anchor" refrences to the       */
|                     /* other members of the output dataset.                          */
|                     /*                                                               */
|                     /*****************************************************************/
|                     Do i = 1 to MigAll.0
|
|                       If MigAll.i = "" Then Iterate
|
|                       If Substr(MigAll.i,1,2) = '/*' Then ,
|                         Queue MigAll.i
```

```
  Else ,
    Queue "<A HREF=#"||Word(MigAll.i,1)||">"||,
           Left(Word(MigAll.i,1),8)||"</A>" Subword(MigAll.i,2)
End

Queue "<br>"

/*********************************************************************/
/*                                                                   */
/* Skip the first entry in the list of member names, $MIGALL,        */
/* and queue up each member into the stream.                         */
/*                                                                   */
/* At begining of the member, create the "Anchor" point that is      */
/* refrenced in the INDEX at the begining.                           */
/*                                                                   */
/*********************************************************************/
Do i = 2 to MCMin.0

  Queue '<A NAME="'||MCMin.i||'">' ,
       '<FONT Color="RED"><b>' ,
        MCMIn.i||"</b></FONT></A>"
  "PIPE MEMBERS '"||SYSDSNAME||"' "||MCMin.i,
      "| STEM MIn."
  Do j = 1 to MIn.0
    Queue Min.j
  End
End

Queue '</PRE>'
Queue '<center><A HREF=#TOP><FONT color=Blue>Top</A></font></center>'
Queue '</BODY></HTML>'
Queue

'PIPE STACK',
    "| > DDName=HTMLOUT"

Return
```

Coupling facility maintenance enhancements

With z/OS V1R9, you can place a coupling facility (CF) into maintenance mode, which can simplify the process of removing all of the structures from a CF. Once a CF is placed in maintenance mode, XCF will not allocate any new structures on that CF. You can then remove the existing structures on the CF without being concerned that new allocations on the CF will occur.

See the topic "Sample procedure for coupling facility maintenance" in *z/OS MVS Setting Up a Sysplex*, SA22-7625 for the commands to use to place a CF into maintenance mode. However, note that in the example shown, the keyword MAINMODE is a typographical error; the correct keyword is MAINTMODE.

Toleration support for coupling facility maintenance enhancements: If you have a sysplex with mixed levels of z/OS, you should apply the fix for APAR OA17685. This will enable z/OS systems at z/OS V1R6 and higher to recognize that a CF is in maintenance mode and prevent those systems from allocating structures on the CF. You must use a z/OS V1R9 system to place a CF into or remove a CF from maintenance mode, but you do not have to wait for the entire sysplex to be migrated to z/OS V1R9 to begin using this function.

Our observations with coupling facility maintenance enhancements: When you place a CF into maintenance mode, the CF remains connected and the CF link paths remain online. If you have previously used the CF drain function of Tivoli System Automation for z/OS, you are used to seeing the CF link paths to the drained CF be taken offline from each z/OS image to prevent further structure allocations on the drained CF. This is no longer necessary with the new CF maintenance mode.

Testing greater than 32 CPU support

With the combination of z/OS V1R9 and System z9™ Enterprise Class (EC), you now can define a single z/OS LPAR image with up to 54 CPUs, which includes System z Application Assist Processors (zAAPs) and System z9 Integrated Information Processors (zIIPs). This function provides flexibility in choosing how to grow: horizontally, with Parallel Sysplex, or vertically, using the greater than 32 CPU support.

Our testing of this new support occurred in two phases:

- **Phase 1: One LPAR with dedicated general purpose CPUs**

  On a System z9 EC, we defined only one z/OS LPAR with 50 dedicated general purpose CPUs, two zAAPs, and two zIIPs. On this image, we ran a high stress level of our IMS, CICS, DB2, WebSphere MQ, z/OS UNIX®, and WebSphere Application Server workloads with a constant number of transactions. We started with 24 general purposes CPUs online and then varied CPUs online—in groups of eight, twice, and a final group of two—to achieve a total of 50 general purpose CPUs, while maintaining the same level of transactions. We preserved the percent of CPU utilization value and found that it scaled as expected.

- **Phase 2: Two LPARs with shared general purpose CPUs**

  We defined two z/OS LPARs, each with 50 shared general purpose CPUs, two shared zAAPs, and two shared zIIPs. On one z/OS LPAR (J80) we ran the high stress IMS, CICS, DB2, and WebSphere Application Server workloads. On the other LPAR (J90) we ran low priority workloads (WebSphere MQ, batch, and z/OS UNIX). On the high stress LPAR, we did a staging run where we gradually increased the number of transactions that the workloads were running. We started monitoring once the workloads started at low levels until they reached stress levels and the CPU utilization for the LPAR was more than 80%. We did this in order to see how WLM and IRD would manage processor resources. When the high stress LPAR reach 90% CPU utilization, we observed that processors were taken away from the low priority LPAR and the weights of both LPARs were adjusted accordingly.

The following are some examples of the RMF™ TM Monitor III screens that show the number of processors:

```
           HARDCOPY     RMF V1R9   CPC Capacity                  Line 1 of 14
Command ===>
Samples: 32      System: J80   Date: 07/26/07  Time: 12.30.00  Range: 60    Sec
Partition:   J80        2094 Model 750
CPC Capacity:    2295   Weight % of Max: 10.0   4h Avg:   32   Group:   N/A
Image Capacity:  2295   WLM Capping %:    0.0   4h Max:  591   Limit:   N/A
Partition  --- MSU ---  Cap  Proc    Logical Util %   - Physical Util % -
             Def   Act  Def   Num    Effect   Total   LPAR  Effect  Total
*CP                         96.0                       0.4    62.9   63.3
J80            0   980  NO  48.0       44.3    44.5    0.2    42.5   42.7
J90            0   468  NO  48.0       21.2    21.2    0.0    20.4   20.4
PHYSICAL                                               0.2            0.2
```

```
*AAP                        4.0                      0.3    91.2   91.4
J80                     NO  2.0       68.9    69.0   0.1    68.9   69.0
J90                     NO  2.0       22.3    22.3   0.1    22.3   22.3
PHYSICAL                                             0.2           0.2

*IIP                        4.0                      0.2    10.4   10.6
J80                     NO  2.0       10.4    10.5   0.0    10.4   10.5
J90                     NO  2.0        0.0     0.0   0.0     0.0    0.0
PHYSICAL                                             0.2           0.2
```

CPU utilization exceeded more than 80% on system J80:

```
            HARDCOPY     RMF V1R9   CPC Capacity                 Line 1 of 14
Command ===>
Samples: 47      System: J80   Date: 07/26/07  Time: 13.31.00  Range: 60    Sec
Partition:   J80        2094 Model 750
CPC Capacity:    2295   Weight % of Max: 10.0   4h Avg:  372   Group:   N/A
Image Capacity:  2295   WLM Capping %:    0.0   4h Max: 1879   Limit:   N/A
Partition  --- MSU ---  Cap  Proc    Logical Util %   - Physical Util % -
             Def   Act  Def   Num    Effect   Total   LPAR  Effect  Total
*CP                        73.0                      0.2    96.4   96.6
J80            0  2029  NO  45.0       98.1    98.2   0.1    88.3   88.4
J90            0   187  NO  28.0       14.5    14.5   0.0     8.1    8.1
PHYSICAL                                             0.1           0.1

*AAP                        4.0                      0.1    94.4   94.5
J80                     NO  2.0       93.4    93.4   0.0    93.4   93.4
J90                     NO  2.0        1.0     1.0   0.0     1.0    1.0
PHYSICAL                                             0.1           0.1

*IIP                        4.0                      0.2    16.0   16.2
J80                     NO  2.0       16.0    16.0   0.0    16.0   16.0
J90                     NO  2.0        0.0     0.0   0.0     0.0    0.0
PHYSICAL                                             0.2           0.2
```

Chapter 5. Using the System z9 Integrated Information Processor (zIIP)

IBM extended its mainframes data serving capabilities, delivering a new roadmap for the future of data serving and information on demand, previewing new DB2 function, and introducing a new specialty engine directed toward data serving workloads.

The new specialty engine, the IBM System z9 Integrated Information Processor (IBM zIIP), is now available on the System z9 Enterprise Class (EC) and System z9 Business Class (BC) servers.

A zIIP is similar in concept to the zSeries Application Assist Processor (zAAP). Like zAAPs; but unlike CPs, ICFs and IFLs, zIIPs can do nothing on their own; they can not perform an IPL and can not run an operating system. zIIPs must operate along with general purpose CPs within logical partitions running z/OS, however they are designed to operate asynchronously with the general purpose CPs to execute selective workloads such as:

- ERP or CRM application serving - For applications, running on z/OS, UNIX, Intel®, or Linux on System z that access DB2 for z/OS V8 on a System z9, through DRDA® over a TCP/IP connection, DB2 gives z/OS the necessary information to have portions of these SQL requests directed to the zIIP.
- Data Warehousing applications – Requests that utilize DB2 for z/OS V8 for long running parallel queries, including complex star schema parallel queries, may have portions of these SQL requests directed to the zIIP when DB2 gives z/OS the necessary information. These queries are typical in data warehousing implementations. The addition of select long running parallel queries may provide more opportunity for DB2 customers to optimize their environment for Data Warehousing while leveraging the unique qualities of service provided by System z9 and DB2.
- Some DB2 for z/OS V8 utilities – A portion of DB2 utility functions used to maintain index maintenance structures (LOAD, REORG, and REBUILD INDEX) that typically run during batch, can be redirected to zIIPs.

This chapter describes what we did to configure and to prepare to exercise and test the zIIP feature on our z9 systems.

Prerequisites for zIIP

The following are prerequisites for zIIP usage:

- z/OS V1R6 with JBB77S9 applied
- z/OS V1R7 with JBB772S applied
- z/OS V1R8
- DB2 V8 with the appropriate maintenance.

More detailed information about all the software and hardware prerequisites can be found in the following PSP buckets:

- Hardware 2094 and 2096 devices buckets.
- z/OS BCP zIIP bucket
- zIIP functional PSP Bucket

Also please contact your local hardware and software representatives for any additional requirements.

Configuring the zIIPs

We configured two zIIPs on all our z/OS images on our System z9 EC and we configured one zIIP on our System z9 BC. When you configure your z/OS logical partitions you simply specify how many logical zIIPs you want to define for each partition, just as you do for the number of standard CPs and zAAPs. When you IPL the system, z/OS determines how many zIIPs are configured and manages an additional dispatcher queue for zIIP-eligible work.

We did the following to configure our zIIPs:

1. Updated the image profile for all our System z9 EC partitions to define two zIIPs to each partition
2. Updated our System z9 BC partition to define one zIIP. Figure 13 shows an example of the image profile for our J80 z/OS image with 2 zIIPs defined:

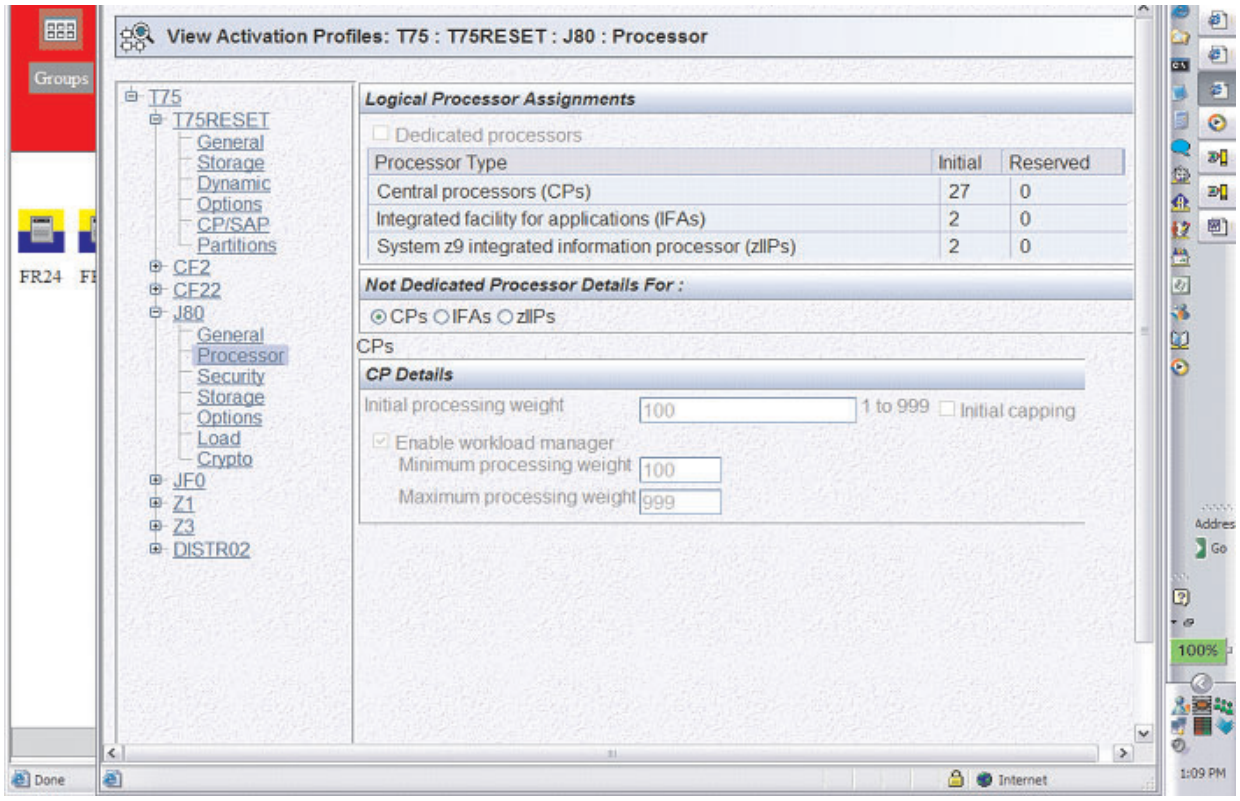


Figure 13. Image profile for our J80 z/OS image with 2 zIIPs defined

3. Deactivated, activated and IPL'd the z/OS partitions to bring the zIIPs online.

   You can use the D M=CPU command to display the status of the zIIPs. The zIIPs appear as an integrated information processor in response to the D M=CPU command.

   Response example for the D M=CPU command on system JH0:

```
-JH0D M=CPU
 IEE174I 13.14.39 DISPLAY M 372
 PROCESSOR STATUS
```

```
 ID  CPU                 SERIAL
 00  +                    01FE2D2096
 01  +                    01FE2D2096
 02  +                    01FE2D2096
 03  +                    01FE2D2096
 04  +A                   01FE2D2096
 05  +I                   01FE2D2096

 CPC ND = 002096.S07.IBM.02.00000002FE2D
 CPC SI = 2096.Z04.IBM.02.000000000002FE2D
 CPC ID = 00
 CPC NAME = K25
 LP NAME = ZOSEJH0    LP ID =  1
 CSS ID  = 0
 MIF ID  = 2
+ ONLINE    - OFFLINE    . DOES NOT EXIST    W WLM-MANAGED
N NOT AVAILABLE

A        APPLICATION ASSIST PROCESSOR (zAAP)
I        INTEGRATED INFORMATION PROCESSOR (zIIP)
CPC ND  CENTRAL PROCESSING COMPLEX NODE DESCRIPTOR
CPC SI  SYSTEM INFORMATION FROM STSI INSTRUCTION
CPC ID  CENTRAL PROCESSING COMPLEX IDENTIFIER
CPC NAME CENTRAL PROCESSING COMPLEX NAME
LP NAME  LOGICAL PARTITION NAME
LP ID    LOGICAL PARTITION IDENTIFIER
CSS ID   CHANNEL SUBSYSTEM IDENTIFIER
MIF ID   MULTIPLE IMAGE FACILITY IMAGE IDENTIFIER
```

Response example for the D M=CPU command on system J80:

```
-D M=CPU
 IEE174I 07.47.11 DISPLAY M 895
 PROCESSOR STATUS
 ID  CPU                 SERIAL
 00  +                    07299E2094
 01  +                    07299E2094
 02  +                    07299E2094
 03  +                    07299E2094
 04  +                    07299E2094
 05  +                    07299E2094
 06  +                    07299E2094
 07  +                    07299E2094
 08  +                    07299E2094
 09  +                    07299E2094
 0A  +                    07299E2094
 0B  +                    07299E2094
 0C  +                    07299E2094
 0D  +                    07299E2094
 0E  +                    07299E2094
 0F  +                    07299E2094
 10  +                    07299E2094
 11  +                    07299E2094
 12  +                    07299E2094
 13  +                    07299E2094
 14  +                    07299E2094
 15  +                    07299E2094
 16  +                    07299E2094
 17  +                    07299E2094
 18  +                    07299E2094
 19  +                    07299E2094
 1A  +                    07299E2094
 1B  +A                   07299E2094
 1C  +A                   07299E2094
 1D  +I                   07299E2094
 1E  +I                   07299E2094

 CPC ND = 002094.S38.IBM.02.0000000C299E
```

```
 CPC SI = 2094.729.IBM.02.00000000000C299E
CPC ID = 00
 CPC NAME = T75
 LP NAME = J80        LP ID =  7
 CSS ID  = 0
 MIF ID  = 7

 + ONLINE    - OFFLINE    . DOES NOT EXIST    W WLM-MANAGED
 N NOT AVAILABLE

 A        APPLICATION ASSIST PROCESSOR (zAAP)
 I        INTEGRATED INFORMATION PROCESSOR (zIIP)
 CPC ND  CENTRAL PROCESSING COMPLEX NODE DESCRIPTOR
 CPC SI  SYSTEM INFORMATION FROM STSI INSTRUCTION
 CPC ID  CENTRAL PROCESSING COMPLEX IDENTIFIER
 CPC NAME CENTRAL PROCESSING COMPLEX NAME
 LP NAME  LOGICAL PARTITION NAME
 LP ID    LOGICAL PARTITION IDENTIFIER
 CSS ID   CHANNEL SUBSYSTEM IDENTIFIER
 MIF ID   MULTIPLE IMAGE FACILITY IMAGE IDENTIFIER
```

Monitoring zIIP utilization:

There is support in RMF to provide information about zIIP utilization. This information is useful to determine if and when you need to add zIIP capacity. For more details about RMF support for zIIPs and new fields on this report, please see *z/OS RMF Report Analysis*, SC33-7991.

Here is an example of our RMF Monitor III, CPC Report that displays the use of zIIP processors (in **bold**) on our System z9 EC images:

```
            HARDCOPY     RMF V1R7   CPC Capacity                  Line 1 of 30
 Command ===>
0Samples: 119     System: J80   Date: 05/24/06  Time: 10.22.00  Range: 120   Sec
0Partition:   J80        2094 Model 729
 CPC Capacity:    1524   Weight % of Max: 10.0       4h MSU Average:   114
 Image Capacity:  1419   WLM Capping %:   ****       4h MSU Maximum:   322
0Partition  --- MSU ---  Cap  Proc    Logical Util %   - Physical Util % -
              Def   Act  Def   Num    Effect   Total   LPAR  Effect  Total
0*CP                                                    5.0    77.9   82.9
 DISTR01        0     1  NO    2.0       0.5     0.6    0.0     0.0    0.0
 DISTR02        0     0  NO    5.0       0.0     0.0    0.0     0.0    0.0
 JF0            0   366  NO   14.0      48.9    49.7    0.4    23.6   24.0
 J80            0   750  NO   23.0      60.6    62.1    1.2    48.1   49.2
 Z1             0    14  NO    8.0       3.2     3.4    0.0     0.9    0.9
 Z3             0    82  NO    8.0      19.2    19.5    0.1     5.3    5.4
 PHYSICAL                                               3.3            3.3

*AAP                                                    2.3    97.6   99.9
JF0                     NO    2.0      32.8    33.0    0.2    32.8   33.0
J80                     NO    2.0      32.7    33.0    0.3    32.7   33.0
Z1                      NO    2.0       0.1     0.3    0.1     0.1    0.3
Z3                      NO    2.0      32.0    32.1    0.1    32.0   32.1
PHYSICAL                                               1.6            1.6

*IFL                                                    0.2     0.0    0.2
PETLVS                  NO    1.0       0.0     0.0    0.0     0.0    0.0
PETLVS2                 NO    1.0       0.0     0.0    0.0     0.0    0.0
PHYSICAL                                               0.2            0.2

*ICF                                                    0.0    99.5   99.5
CF2                           3.0      99.6    99.6    0.0    74.7   74.7
CF22                          1.0      99.4    99.4    0.0    24.9   24.9
PHYSICAL                                               0.0            0.0

*IIP                                                   4.0    59.8   63.7
```

```
JF0                  NO    2.0      17.2    17.6    0.4     17.2    17.6
J80                  NO    2.0      42.6    43.3    0.8     42.6    43.3
PHYSICAL                                            2.8             2.8
```

SMF type 70.1, 72.3, 79.1 and 79.2 records contain new fields with zIIP measurements. There are also new fields in SMF type 30 records to indicate the amount of time spent in zIIP work as well the amount of time spent executing zIIP eligible work on standard processors. *z/OS MVS System Management Facilities (SMF)*, SA22-7630 can give you details on the new fields.

SDSF also provides information about system zIIP utilization as well as enclave zIIP utilization. New columns on the DA display and the Enclave display have been added to provide this information. For more details about these new fields for SDSF please see *z/OS SDSF Operation and Customization*, SA22-7670.

Here is one example for the SDSF enclave display that shows zIIP utilization on our z9 EC systems:

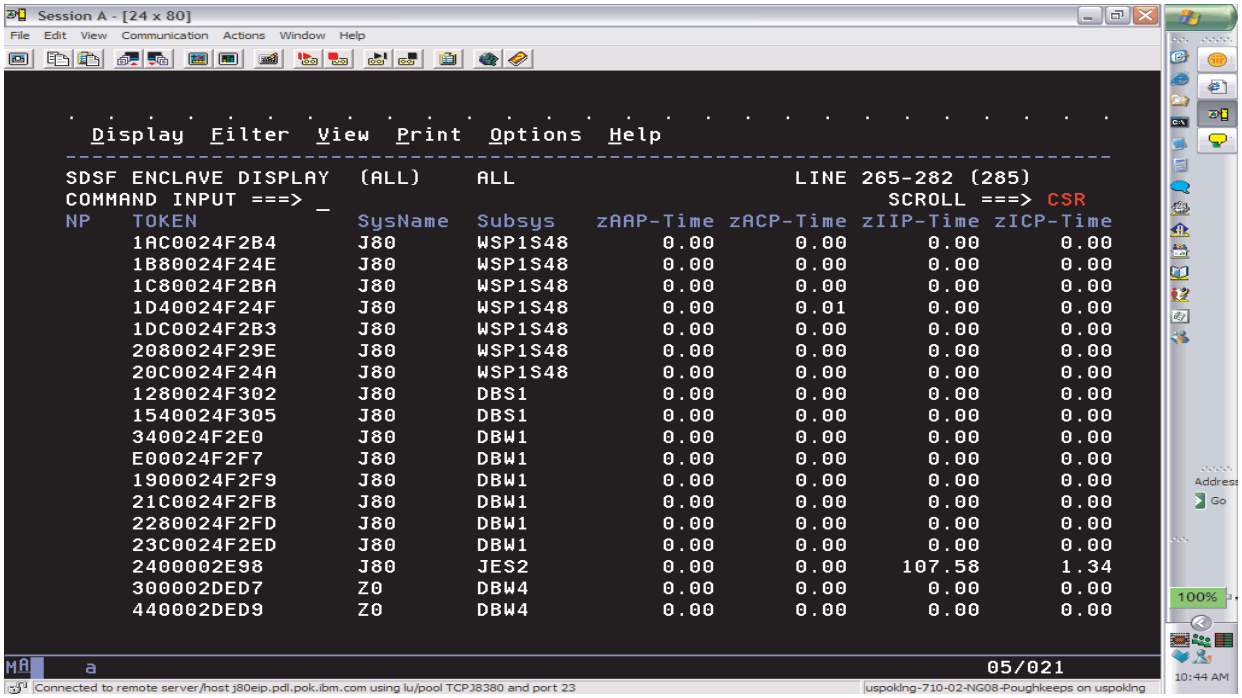


Figure 14. SDSF display showing zIIP utilization

Workloads that exercise the zIIP processors

The System z9 Integrated Information Processor (zIIP) is designed so that specific types of DB2 programs or utilities can negotiate with z/OS to have a portion of their enclave Service Request Block (SRB) work redirected from the general purpose Central Processor (CP) over to the zIIP, thereby freeing the CP for other tasks.

Those types of work which do not utilize SRBs, such as stored procedures and user-defined functions, are not eligible to offload work to the zIIP.

Currently, there are basically three situations or scenarios that may benefit from having a portion of their SQL requests redirected to the zIIP; they include the following:

1. Applications running on z/OS, UNIX, Intel, or Linux on System z that access DB2 via DRDA over a TCP/IP connection.

   To test offloading portions of SQL requests using DRDA access over a TCP/IP connection to zIIP, we employed the use of the IBM Trade Performance Benchmark Sample for WebSphere Application Server V6.0 (or simply the Trade 6 workload), which may be obtained from the following website:

   `https://www.software.ibm.com/webapp/iwm/web/preLogin.do?source=trade6`

   Logon (or register if you are a new user), download tradeInstall.zip (1.7MB), and refer to the *Trade Technology* document (tradeTech.pdf) located in the install package for general information regarding Trade 6.

   During our testing, we were able to drive substantial zIIP utilization using the Trade 6 workload and were able to monitor it via RMF Monitor III.

2. Requests that utilize DB2 for long running complex parallel queries, such as star schema parallel queries.

   For this particular scenario, we made use of the following star join query which was executed after having enabled star schema parallelism:

```
SELECT COUNT(*) FROM
              ADMF001.TBFACT1 F,
              ADMF001.TBDIMN01 D1,
              ADMF001.TBDIMN02 D2,
              ADMF001.TBDIMN03 D3,
              ADMF001.TBDIMN04 D4,
              ADMF001.TBDIMN05 D5
  WHERE
    F.TIME_CLOSED_KEY  = D1.TIME_CLOSED_KEY   AND
    F.TOD_KEY          = D2.TOD_KEY           AND
    F.RECEIVED_VIA_KEY = D3.RECEIVED_VIA_KEY  AND
    F.CASE_KEY         = D4.CASE_KEY          AND
    F.CUSTOMER_KEY     = D5.CUSTOMER_KEY      AND
F.TIME_CLOSED_KEY  = 182;
```

   We noted some activity being redirected to the zIIP, but not a great deal. Note that even though star schema parallelism has been enabled and a zIIP is available for use, the DB2 Optimizer can decide that the optimal path is not to use star join, thus bypassing the zIIP. The optimizer's focus is not whether the query can take advantage of zIIP offload or not, but rather choosing the lowest cost access path.

3. Some DB2 utilities used in the maintenance of index structures that are normally executed in batch, such as the LOAD, REORG, and REBUILD INDEX utilities.

   Testing the offloading of portions of the DB2 LOAD, REORG, and REBUILD INDEX utilities to zIIP entailed the creation of a batch workload comprised of three jobs, each of which performs a task specific to zIIP testing:

   LOAD
   : Reloads tables

   RBLDINDX
   : Rebuilds indexes

   REORG
   : Reorgs tables

   The jobs are currently chained together with LOAD executing first; LOAD then calls RBLDINDX, which in turn calls REORG. If desired, for continuous operation REORG can be set to call LOAD again. The three jobs together take about a half hour to complete. Of the three scenarios mentioned, this particular

one redirected more work to the zIIP than the star schema parallel queries but less than the Trade 6 workload utilizing DRDA over TCP/IP connections.

OMEGAMON XE for z/OS 3.1.0 zIIP SUPPORT

We recently installed OMEGAMON® XE for z/OS 3.1.0 into PET. To learn more about OMEGAMON XE for z/OS 3.1.0 go to:

```
http://www-306.ibm.com/software/tivoli/products/omegamon-xe-zos/
```

If you already have OMEGAMON XE for z/OS 3.1.0 installed you will need the following support to enable the zIIP support:

```
OB550:  UA27609 (APAR OA15898)
M2550:  UA27610 (APAR OA15899)
M5310:  UA27611 (APAR OA15900)


OP360 TEP: 3.1.0-TIV-KM5-IF0001
ITM6.1 TEP 3.1.0-TIV-KM5-ITM-IF0001
```

We were the first Plex with zIIPs to actually verify and use the OMEGAMON XE for z/OS 3.1.0 zIIP support. To access the OMEGAMON Classic support for zIIP, select 'C CPU' from the OMEGAMON MAIN MENU. See Figure 15. zIIP is represented by **IIP**.

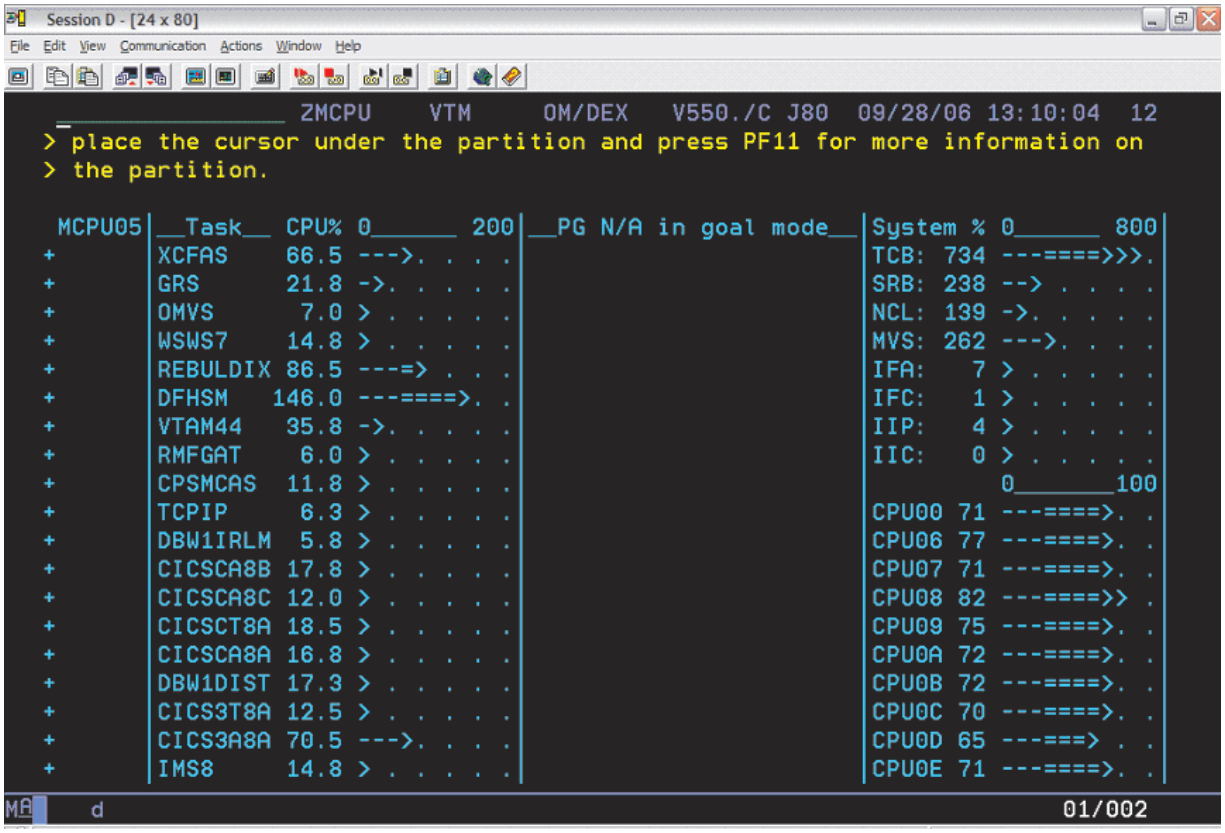


Figure 15. OMEGAMON ZMCPU screen

From your TEP server, the OMEGAMON XE for z/OS zIIP can be found in the predefined workspace 'System CPU Utilization'. Figure 16 on page 68 and Figure 17 on page 69 show the TEP OMEGAMON XE for z/OS 'System CPU

Utilization' workspace:

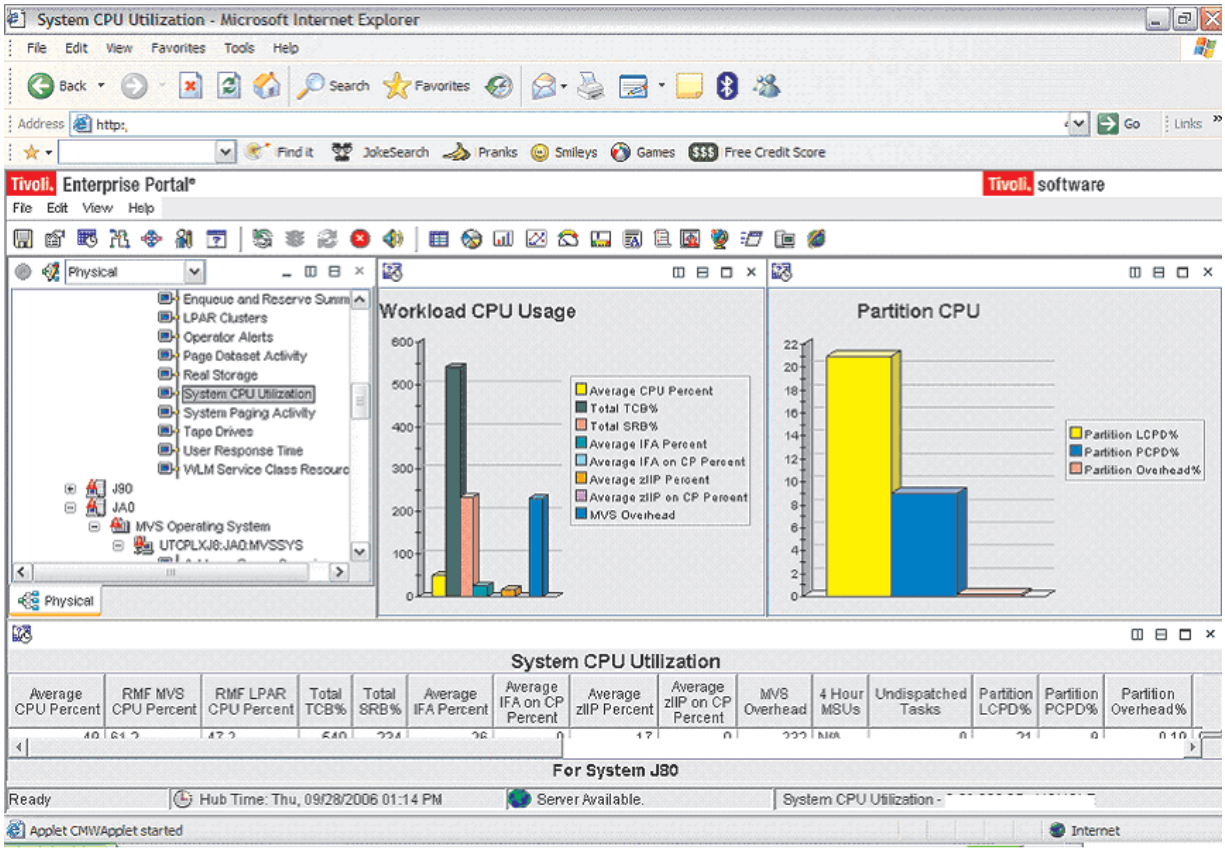


Figure 16. OMEGAMON System CPU Utilization 1

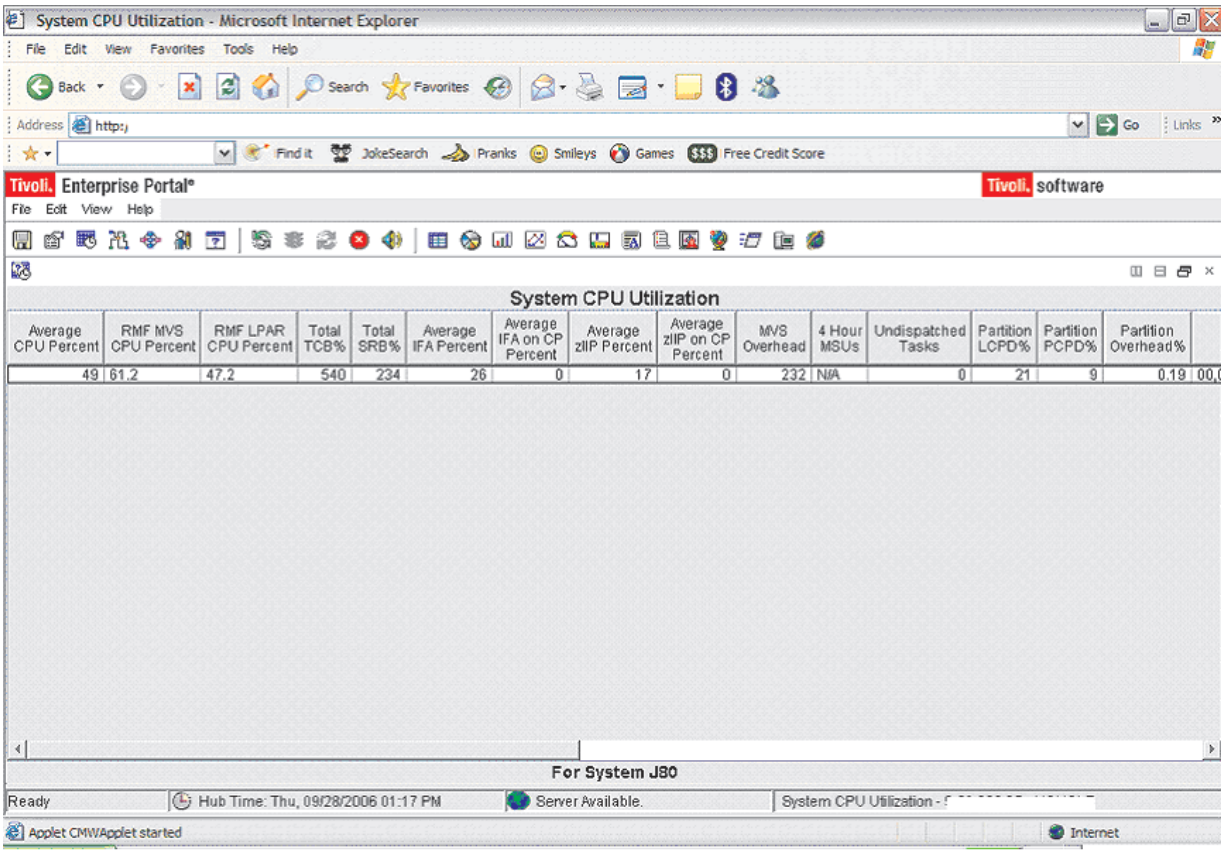


Figure 17. OMEGAMON System CPU Utilization 2

From your TEP server, the OMEGAMON XE for z/OS zIIP support can also be found in the predefined workspace 'Address Space Overview' as seen in Figure 18 on page 70.

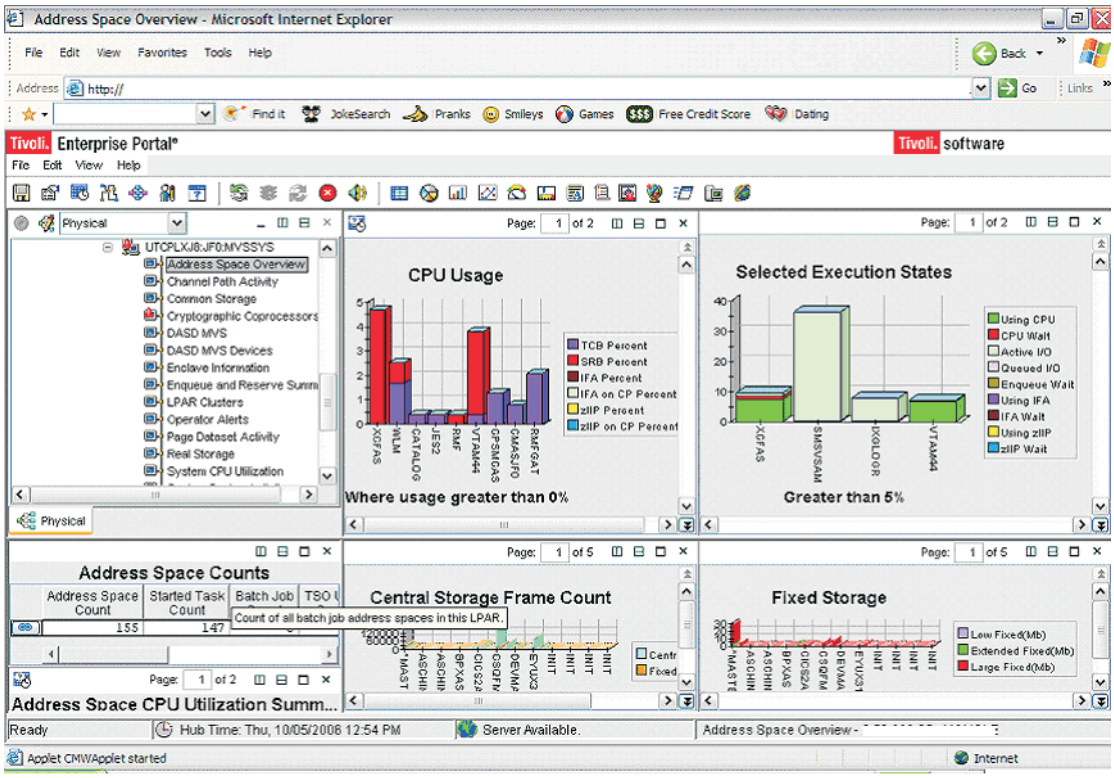


Figure 18. OMEGAMON Address Space Overview

zIIP Assisted IPSec

Beginning with z/OS V1R8, the IBM System z9 Integrated Information Processor (zIIP) can be used to handle much of the CPU-intensive processing involved in the IPSec Authentication Header (AH) and Encapsulating Security Payload (ESP) protocols. (For details, see IBM announcement 107-190, dated 18 April 2007.)

The new zIIP Assisted IPSec function allows z/OS Communications Server to interact with z/OS Workload Manager to have its enclave service request block (SRB) work directed to zIIP. Since our System z9 CPC already had zIIP processors configured, we changed our existing IPSec deployment to use the zIIP processors to reduce the amount of general purpose CP consumption imposed by our IPSec workloads.

Related information:: It is beyond the scope of this discussion to provide installation and configuration information related to deploying IPSec on z/OS. For information about deploying IPSec on z/OS, see the z/OS Communications Server library and SG24-7342, from the IBM Redbooks library at www.ibm.com/redbooks/.

Implementing the zIIP Assisted IPSec support required that we install Communications Server TCP/IP APAR PK40178. Once we installed this enabling APAR, we performed one required configuration change and completed one strongly recommended task, as follows:

1. We added the GLOBALCONFIG ZIIP IPSECURITY statement to our TCP/IP Profile configuration.

This configuration statement is required to cause Communications Server to request that z/OS direct the IPSec enclave SRB processing to the available zIIPs. The default for this configuration statement is GLOBALCONFIG ZIIP NOIPSECURITY.

2. We created an independent enclave so that IPSec traffic would be classified and managed, within z/OS Workload Manager, differently than its owning address space (that is, it can be classified and managed differently than the TCP/IP address space). This task is optional; however, creating this enclave is strongly recommended.

We already had zIIPs in use in our environment. Thus, we already had APAR OA20045 applied. This APAR allows zIIP tuning controls to be specified in the IEAOPT*xx* member of PARMLIB. For our environment, we ran with the default values for the following two parameters:

IIPHONORPRIORITY
Specifying IIPHONORPRIORITY=YES allows the zIIP-eligible workload to run on standard CPs if zIIP work is not completed in a reasonable time period (see the ZIIPAWMT parameter). This is the default and recommended value. Specifying IIPHONORPRIORITY=NO disallows any zIIP-eligible work from running on standard CPs.

ZIIPAWMT
ZIIPAWMT controls how aggressive z/OS will be in requesting help from other zIIPs or CPs when IIPHONORPRIORITY=YES and all zIIPs are busy. We ran with the default value of 12 milliseconds.

While we noticed a significant benefit with the deployment of this solution, it is beyond the scope of this report to cite the performance benefits associated with our zIIP Assisted IPSec deployment. However, you can find the configuration requirements for zIIP Assisted IPSec, capacity planning information, and zIIP IPSec performance data at www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP100988.

Chapter 6. Migrating CICS

The following topics describe our experiences with CICS migration:

- "Migrating to CICS Transaction Gateway V6.1"

Migrating to CICS Transaction Gateway V6.1

We migrated our CICS Transaction Gateway (CICS TG) setups from V6.0 to V6.1. Our migration was very simple and straight-forward and our CICS TG continues to run solidly.

The basic setups for CICS TG V6.1 are much the same as for V6.0. See the "Migration to CICS Transaction Gateway V6.0" topic in our December 2005 test report for full details about our CICS TG V6 setups.

Migrate CICS TG daemon to V6.1 first

CICS Transaction Gateway supports communication with CICS TG resource adapters of the same level or an earlier level. To maintain this compatibility, we migrated our CICS TG daemons to V6.1 before the client side code (such as WebSphere Application Server resource adapters).

Do not mix CICS TG in WebSphere Application Server

Both WebSphere Application Server V6.0 and V6.1 support the resource adapter provided by CICS TG V6.0 or V6.1, so there is greater flexibility here for when you choose to migrate your application server environment. However, you cannot have multiple levels of CICS TG resource adapters within your WebSphere Application Server nodes. Uninstall any existing version of the CICS TG resource adapter prior to installing version 6.1.

CICS TG references

See the following resources for more information about CICS Transaction Gateway:

- CICS Transaction Gateway home page at www.ibm.com/software/htp/cics/ctg/
- Documentation for CICS TG V6.1 at www.ibm.com/software/htp/cics/ctg/library/

Chapter 7. Migrating to DB2 Version 9.1

This chapter addresses the processes and experiences encountered during the migration of the Integration Test production 3 way DB2® data sharing group DBSG from DB2 Version 8 to Version 9.1 (composed of members DBS1, DBS2, DBS3).

We used the *DB2 Installation Guide, (GC18-9846-00)* for our migration. Whenever we reference a **Migration Step** in **bold** in this chapter, we are referencing the same migration steps that are in the *DB2 Installation Guide*.

Migrating DB2 on z/OS requires common known administration skills on zSeries(z/OS) platform. This chapter is organized in the following sections:

Migration considerations

Before you migrate to DB2 Version 9, note the following points:

- Migrations to DB2 Version 9 are only supported from subsystems currently running DB2 Version 8; unpredictable results can occur if a migration is attempted from another release of DB2.
- **Migration Step 24** is an optional step that is used to verify the DB2 Version 9 subsystem after it is in compatibility mode. For this step, only the following selected Version 8 IVP jobs can be executed:
  1. Version 8 phase 2 IVP applications
     a. DSNTEJ2A - All steps except the first two
     b. DSNTEJ2C - Only step PH02CS04, statement RUN PROGRAM(DSN8BC3) PLAN(DSN8BH61), is to be executed
     c. DSNTEJ2D - Only step PH02DS03, statement RUN PROGRAM(DSN8BD3) PLAN(DSN8BD61), is to be executed
     d. DSNTEJ2E - Only step PH02ES04, statement RUN PROGRAM(DSN8BE3) PLAN(DSN8BE61), is to be executed
     e. DSNTEJ2F - Only step PH02FS03, statement RUN PROGRAM(DSN8BF3) PLAN(DSN8BF61), is to be executed
     f. DSNTEJ2P - Execute step PH02PS05
  2. Version 8 phase 3 IVP applications
     a. ISPF-CAF applications, with the exception of DSNTEJ3C and DSNTEJ3P.

Note: If you want to run these IVPs as part of the verification of DB2 Version 9 compatibility mode, they must first be run under Version 8 in their entirety before you start the Version 9 migration process and must remain available for use after you complete the migration to Version 9 compatibility mode.

- Examining "Migration Considerations" of the *DB2 Installation Guide, (GC18-9846-00)*, the following items are of particular interest:
  - Global temporary tables require a 32K buffer pool.
  - Declared global temporary tables and static scrollable cursor result tables require a table space with a 32-KB page size because 8-KB and 16-KB page sizes are not supported for table spaces that are created in the work file database.
  - Declared global temporary tables need a 32-KB table space in the work file database.
  - There are changes to the format of the BSDS. To support up to 10000 data sets per copy for archive logs and 93 data sets per copy for active logs, the BSDS must be converted using job DSNTIJUZ
  - The work file database is the only temporary database. The TEMP database is no longer used by DB2
  - If the application uses GROUP ATTACH, then the GROUP ATTACH process is randomized so that all members running on the same z/OS image have an equal chance of getting attach.
  - Changes in the BIND PACKAGES and BIND PLAN defaults changed from CURRENTDATA YES to NO.
- Functions that are no longer supported:
  - Java stored procedures no longer run in resettable JVMs.
  - DB2-established stored procedure address spaces are no longer supported. Stored procedures must be moved to a WLM environment.
  - JBDC/SQLJ Driver for OS/390 and z/OS is no longer supported. All procedures need to be modified to work with the IBM DB2 Driver for JDBC and SQLJ.
  - Simple table spaces are no longer supported. The default is segmented.
- During the migration of the first member of a data sharing group to DB2 Version 9, other members of the data sharing group can be active, although they can experience delays or time-outs when accessing catalog objects as these objects might be locked because of the migration process. Upon completion of the migration process for all data sharing group members, you must update TSO and CAF logon procedures to reference the DB2 Version 9 libraries exclusively.
- The Administrative Task Scheduler (ATS) as currently implemented in DB2 for z/OS is the first piece of tooling infrastructure for our next-gen Web-and Eclipse based tooling.

  The subsystem parameter ADMTPROC, in macro DSN6SPRM, saves the start procedure name of the Admin Scheduler that is associated with the DB2 subsystem. ADMTPROC cannot be updated online. Whenever DB2 starts up, it starts the Admin Scheduler that is specified in ADMTPROC, if it is not up yet. In addition, every time DB2 starts or stops, it posts an event to the Admin Scheduler so that the Admin Scheduler can execute tasks that depend on these events.

  Reference material:
  - *DB2 Installation Guide, (GC18-9846-00)*
  - *DB2 Version 9.1 for z/OS Administration Guide (SC18-9840-00)*

– *DB2 Version 9.1 for z/OS Application Programming and SQL Guide (SC18-9841-00)*

Premigration activities

Before migrating to DB2 Version 9, application of the fallback SPE to all members of the Version 8 data sharing group is necessary.

Also, ensure that the size of the work file database is sufficiently large enough to support the sorting of indexes when migration job DSNTIJTC is run.

After making a backup of the current logon procedure in use, we updated the procedure to reflect the following DB2 Version 9 concatenations before invoking the DB2 installation CLIST:

- DB2.DB2910.SDSNSPFM was concatenated to ISPMLIB.
- DB2.DB2910.SDSNSPFP was concatenated to ISPPLIB.
- DB2.DB2910.SDSNSPFS was concatenated to ISPSLIB.
- DB2.DB2910.SDSNSPFT was not concatenated to ISPTLIB, as DB2 online help was not installed.

After we logged on with the updated logon procedure, we invoked the installation CLIST DSNTINST from the ISPF Command Shell by entering the following command:

```
ex 'DB2.DB2910.SDSNCLST(DSNTINST)'
```

We filled in the first panel DSNTIPA1 as shown in Figure 19 on page 78.

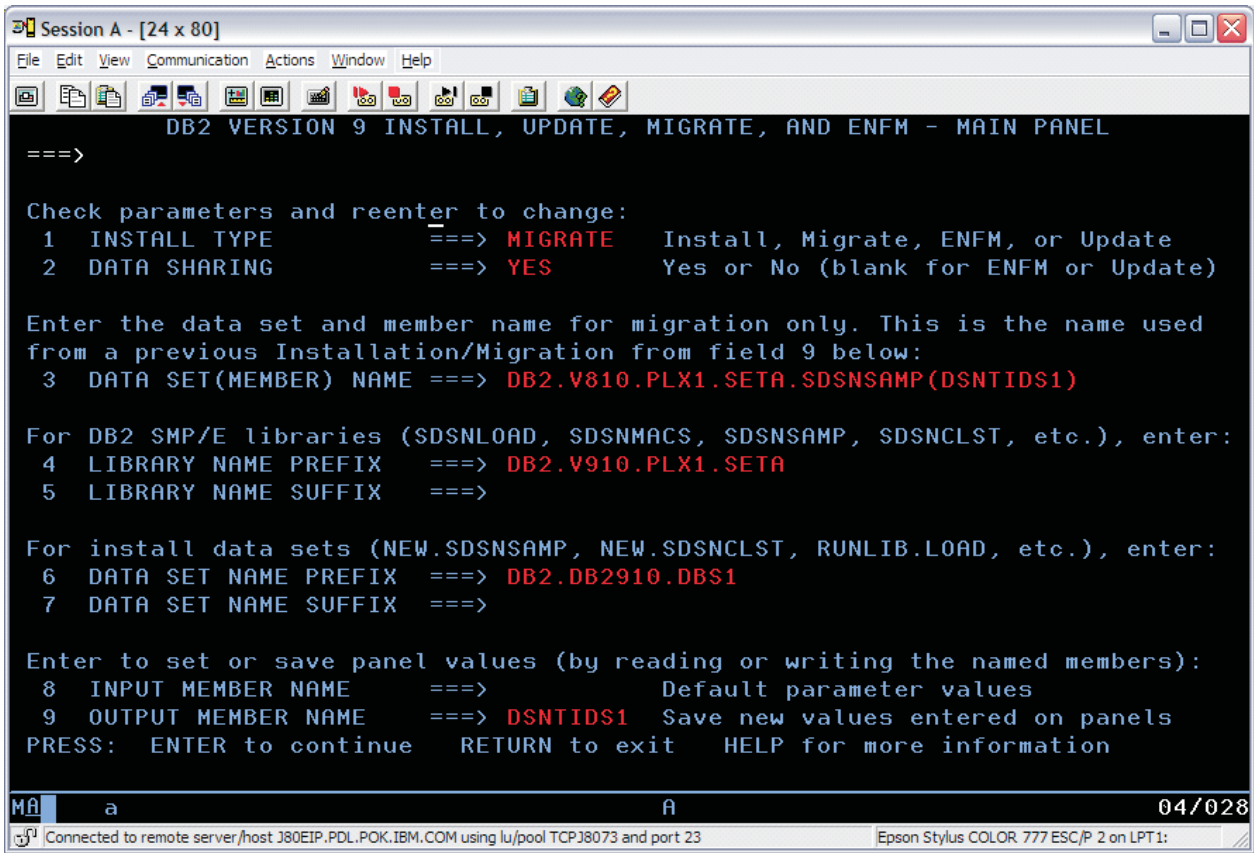


Figure 19. DSNTIPA1

When we pressed enter, the pop-up screen DSNTIPP2 appeared as shown in Figure 20 on page 79.

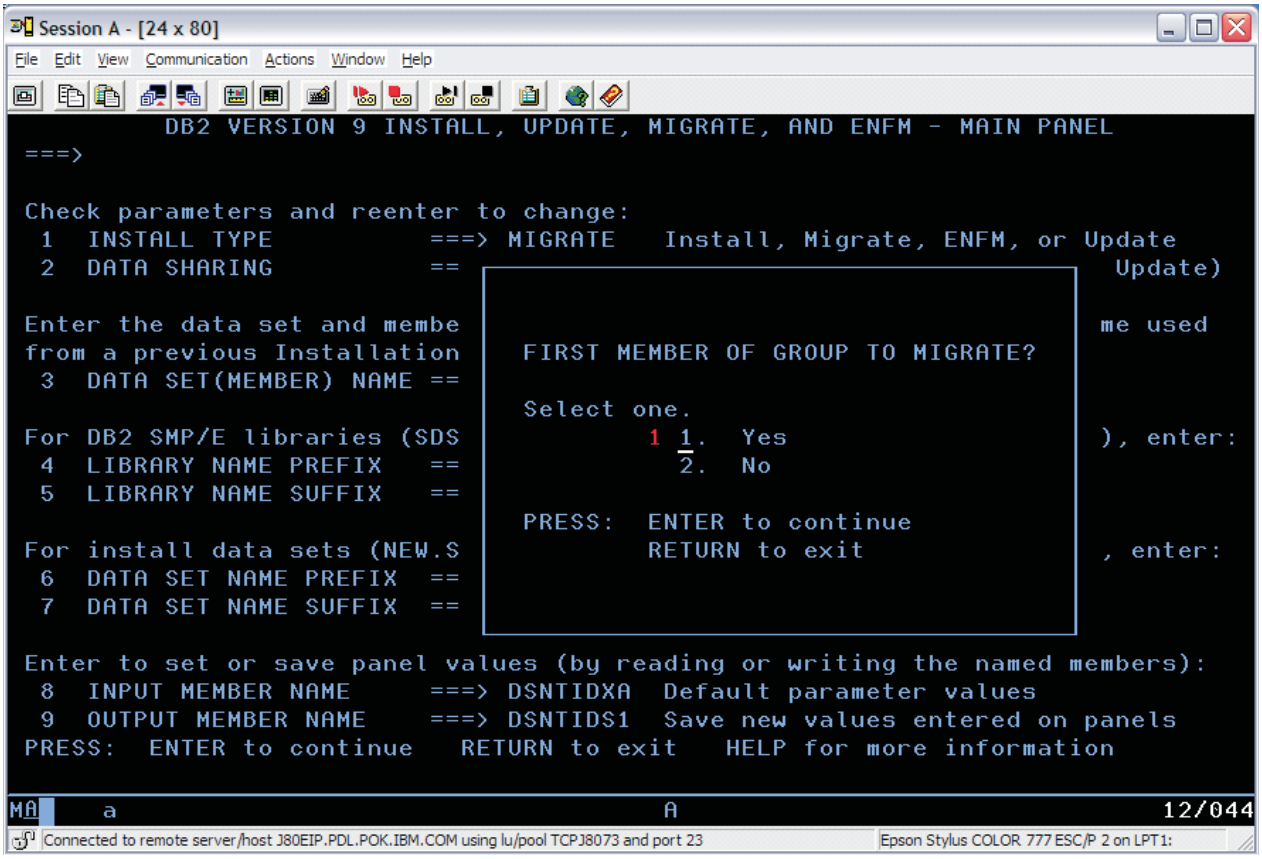


Figure 20. DSNTIPP2

We entered '**1**' to reflect that this was the first member of the data sharing group to be migrated to DB2 Version 9. From this point, we scrolled through the panels and accepted the existing values; upon completion, we placed the tailored JOBS in DB2.DB2910.DBS1.NEW.SDSNSAMP and PROCS in DB2.DB2910.DBS1.NEW.SDSNTEMP as shown in the following:

```
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJMV)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJIN)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJTC)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJTM)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJIC)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJVC)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJSG)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJOS)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJEX)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJGF)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJFT)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJPD)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNTEMP(DSNU)',     CLIST
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNTEMP(DSNH)',     CLIST
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNTEMP(DSNHC)',    CLIST
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNTEMP(DSNEMC01)', CLIST
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJCX)', MIGRATE JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJRI)', INSTALL JCL
IKJ52338I DATA SET 'DB2.V910.PLX1.SETB.SDSNSAMP(DSNTIJRI)' NOT LINE NUMBERED, USING NONUM
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJFV)', FALL BACK JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS1.NEW.SDSNSAMP(DSNTIJUZ)', INSTALL JCL
```

Migrating the first member to compatibility mode

After we reviewed the topics outlined in **Migration Step 1**, we made the following observations:

- Ensured that the IVP jobs and sample database objects for DB2 Version 8 are still available for use. Failure to do so will prevent verifying that a successful migration to DB2 Version 9 compatibility mode has been made.
- Ensured that no utilities are running before migrating to DB2 Version 9. When the migration to Version 9 compatibility mode has been completed, any outstanding utilities that were started under Version 8 cannot be restarted or terminated under Version 9.

Migration Step 2 concerns the optional step of executing DSN1CHKR to verify the integrity of the DB2 directory and catalog table spaces that contain links or hashes. We chose not to run this JOB at this time since we had active applications running on the DB2 V8 members during migration.

Finally, to ensure that there were no STOGROUPs defined with both specific and nonspecific volume ids, we ran the following query:

```
SELECT * FROM SYSIBM.SYSVOLUMES V1
      WHERE VOLID <> '*' AND
      EXISTS (SELECT * FROM SYSIBM.SYSVOLUMES V2
                WHERE V1.SGNAME = V2.SGNAME AND V2.VOLID='*');
```

The query did not return any rows.

Migration Step 3 is an optional step to determine which plans and packages are to be rendered not valid as a result of migrating to DB2 Version 9. To accomplish this, we ran the following queries:

```
SELECT DISTINCT DNAME
                 FROM SYSIBM.SYSPLANDEP
                 WHERE BNAME IN('DSNVVX01','DSNVTH01') AND
                       BCREATOR = 'SYSIBM' AND
                       BTYPE IN ('I','T')
                 ORDER BY DNAME;
               SELECT DISTINCT COLLID, NAME, VERSION
                 FROM SYSIBM.SYSPACKDEP, SYSIBM.SYSPACKAGE
                 WHERE BNAME IN('DSNVVX01','DSNVTH01')
                   AND LOCATION = ' '
                   AND BQUALIFIER = 'SYSIBM'
                   AND BTYPE IN ('I','T')
                   AND COLLID = DCOLLID
                   AND NAME = DNAME
                   AND CONTOKEN = DCONTOKEN
                 ORDER BY COLLID, NAME, VERSION;
```

The first query did not produce any rows, while the second generated the results shown in Figure 21 on page 81.

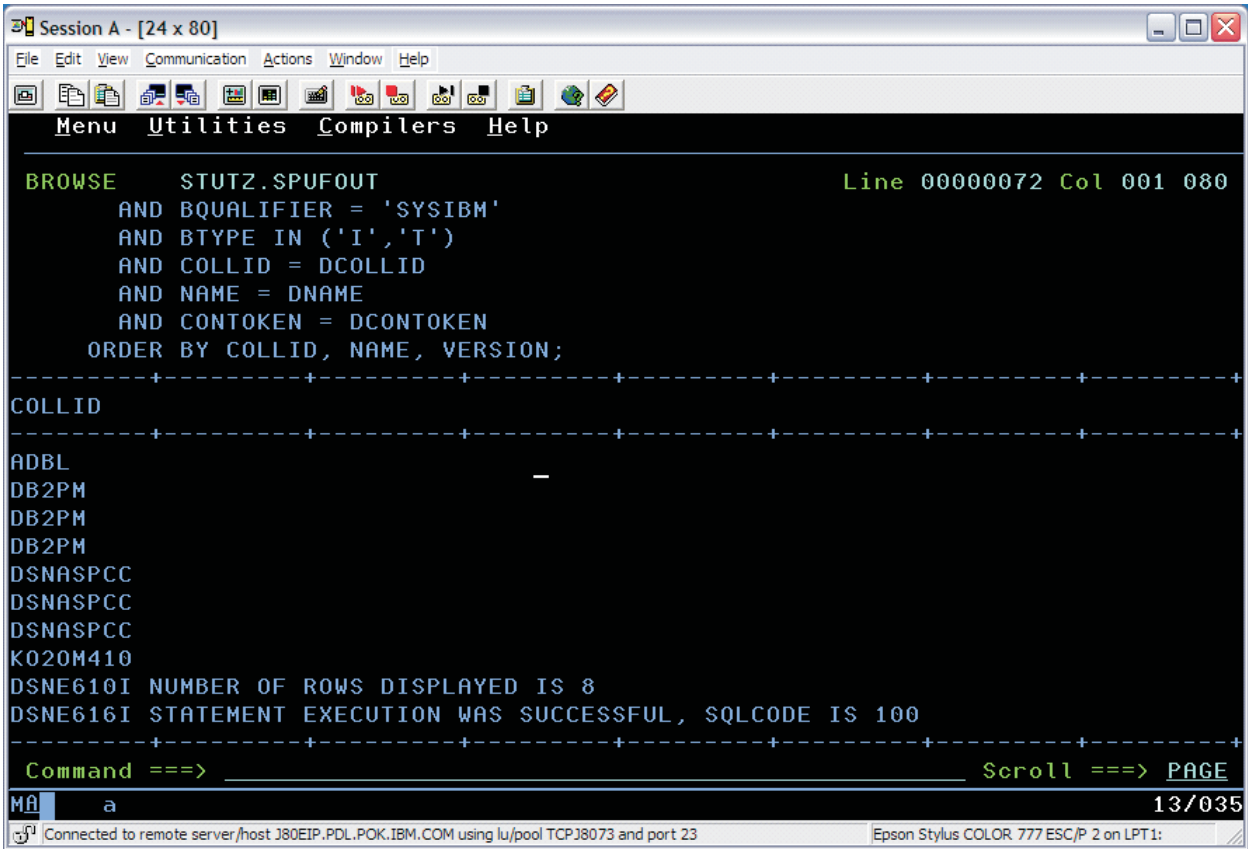


Figure 21. Query output to find packages that will be invalidated when migrating to DB2 Version 9

Migration Step 4 is another optional step to check for consistency between catalog tables through running the queries contained in DB2.DB2910.SDSNSAMP(DSNTESQ). There are a total of 65 queries contained in this data set. We used the data set as input to SPUFI, it ran with no inconsistencies.

Migration Step 5 addresses performing an image copy of the catalog and directory in case of fallback. The *DB2 Installation Guide, (GC18-9846-00)*, recommends using the Version 9 job DSNTIJIC . We followed the recommendation.

Migration Step 6 addresses the following steps necessary to connect DB2 to TSO:

- **Making DB2 load modules available to TSO and batch users -** . Since we run with multiple versions of DB2 V8 and V9 we are using symbolics and extended aliases for TSO and Batch users. SDSNEXIT:

```
DB2.DB2910.DBSG.SDSNEXIT , SDSNLOAD: DB2.DBSG.SDSNLOAD
                  SYMBOLIC:   DB2.&DBSGVER..&DB2PLEX..&DBSGSET..SDSNLOAD
                 EXTENDED ALIAS:
  DEFINE ALIAS (                                          -
  NAME(DB2.DBSG.SDSNLOAD)                                -
  SYMBOLICRELATE(DB2.&DBSGVER..&DB2PLEX..&DBSGSET..SDSNLOAD))
```

- **Making DB2 CLISTs available to TSO and batch users: DSNTIJVC** - Our logon proc DB29PLX1 must again be updated to add DB2.DB2910.NEW.SDSNCLST to the SYSPROC concatenation. We had to do this after we ran the installation job DSNTIJVC (the job that merges tailored CLISTs from prefix.NEW.SDSNTEMP with unchanged CLISTs from prefix.SDSNCLST and places the resulting set of CLISTs in the newly created data set

prefix.NEW.SDSNCLST). Since we currently use fixed-block CLIST libraries (use the SYSPROC concatenation in logon proc DB29PLX1), we had to modify DSNTIJVC as follows:

- Changed the SYSIN DD to DUMMY
- Changed the allocation of prefix.SDSNCLST to match the data control block (DCB) attributes of our other CLIST libraries; this was accomplished by replacing the DCB attributes for DSNTIVB.SYSUT2 with **DCB=*.SYSUT1**.

After DSNTIJVC successfully ran, we updated logon proc DB29PLX1 to add DB2.DB2910.NEW.SDSNCLST to the SYSPROC concatenation.

- **Making panels, messages, and load modules available to ISPF and TSO -** We previously added SDSNSPFP, SDSNSPFM, and SDSNSPFS to the ISPF concatenations. In addition, we updated the logon proc DB29PLX1 to reflect the concatenation of the DB2 English DB2I panels as follows:
  - DB2.DB2910.SDSNPFPE concatenated to ISPPLIB.

Because IMS and CICS connections to DB2 had previously been established, we skipped **Migration Step 7** and **Migration Step 8**.

Migration Step 9 instructs us to stop all DB2 V8 activity or else fallback procedures may fail; prior to stopping data sharing member DBS1, we insured that there were no incomplete utilities (**-DBS1 DISPLAY UTILITY(*)**), and that no databases were in restrict or advisory status (**-DBS1 DISPLAY DATABASE(*) SPACE(*) RESTRICT** and **-DBS1 DISPLAY DATABASE(*) SPACE(*) ADVISORY**, respectively); DBS1 was then brought down.

We skipped optional **Migration Step 10** (**Back Up your DB2 Version 8 volumes**) and performed **Migration Step 11**, which defines DB2 initialization parameters through DSNTIJUZ. After modifying this job by removing the SMP/E step, we submitted it and it ran successfully; expect a return code of 888 if the BSDS has already been converted to the new format.

Special considerations for (Migration Step 11): Step DSBTCNVB converts your BSDS to a New Format. This can be accomplished prior to the migration .We made a decision to convert to the new format prior to the migrations. Following is the DSBTCNVB step:

```
CONVERT THE BSDS TO NEW FORMAT
  NOTE: RC = 888 MEANS BSDS WAS ALREADY CONVERTED
```

As subsystem security had already been established, we skipped **Migration Step 12**.

Migration Step 13 defines DB2 V9 to MVS. We examined job DSNTIJMV to see which modifications to the MVS environment were required; they were implemented accordingly. DSNTIJMV performs the following actions:

- Updates IEFSSNxx, APF, and linklist members, which were deemed not necessary as they had been UPDATED manually RENAME renames the current DB2 procedures in proclib. We skipped this step, however. The DB2 startup procs for DBS1 are renamed manually (see below).
- Step DSNTIPM adds catalogued procedures to proclib; however rather than directing the output of this step to SYS1.PROCLIB, we directed it to a newly created data set, DB2.DB2910.DBSG.PROCLIB.

We renamed the startup procs for DBS1 that reside in PET.PROCLIB (as per the RENAME step of DSNTIJMV). Next, we copied the new V9 startup procs for DBS1 from DB2.DB2910.DBSG.PROCLIB.

For **Migration Step 14**, we successfully ran job DSNTIJIN to define system data sets.

For **Migration Step 15**, we ran the last two steps of job DSNTIJEX to assemble and link edit the access control authorization exit DSNXSXAC and user exit routine DSNACICX (invoked by stored procedure DSNACICS). We skipped the first and second steps that are used to assemble and link edit the signon (DSN3@SGN) and identify (DSN3@ATH) exits because they were not previously implemented.

Because we had previously IPLed the system to pick the V9 early code, we skipped **Migration Step 16**.

Member DBS1 of data sharing group DBSG was then started (**Migration Step 17**) successfully. As the DISPLAY GROUP command shows in the example below, the level of the data sharing group DBSG is now 910 and it is in compatibility mode (MODE(C)); the DB2 level of DBS1 reflects that it is now running DB2 Version 9 code.

```
RESPONSE=J80
 DSN7100I  @DBS1 DSN7GCMD
 *** BEGIN DISPLAY OF GROUP(DSNDBSG ) GROUP LEVEL(910) MODE(C)
                   PROTOCOL LEVEL(2)  GROUP ATTACH NAME(DBSG)
 ------------------------------------------------------------------
 DB2                                    DB2 SYSTEM    IRLM
 MEMBER   ID  SUBSYS CMDPREF   STATUS   LVL NAME      SUBSYS IRLMPROC
 -------- --- ----   --------  -------- --- --------  ----   --------
 DBS1       1 DBS1   @DBS1     ACTIVE   910 J80       IRS1   DBS1IRLM
 DBS2       2 DBS2   @DBS2     ACTIVE   810 JB0       IRS2   DBS2IRLM
 DBS3       4 DBS3   @DBS3     ACTIVE   810 JF0       IRS3   DBS3IRLM
 ------------------------------------------------------------------
```

Migration Step 18. We submitted and ran DSNTIJTC successfully. The job periodically issued message DSNU777I in SYSPRINT to indicate migration progress, as shown in the message DSNU777I which displays CATMAINT progress:

```
DSNU1044I PROCESSING SYSIN AS EBCDIC
DSNU050I            CATMAINT UPDATE
DSNU750I       CATMAINT UPDATE PHASE 1 STARTED
DSNU777I       CATMAINT UPDATE STATUS - VERIFYING CATALOG IS AT CORRECT LEVEL FOR MIGRATION.
DSNU777I       CATMAINT UPDATE STATUS - BEGINNING MIGRATION SQL PROCESSING PHASE.
DSNU777I       CATMAINT UPDATE STATUS - BEGINNING ADDITIONAL CATALOG UPDATES PROCESSING.
DSNU777I       CATMAINT UPDATE STATUS - UPDATING DIRECTORY WITH NEW RELEASE MARKER.
DSNU752I       CATMAINT UPDATE PHASE 1 COMPLETED
DSNU010I       UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
```

Migration Step 19 is an optional step to ensure that there are no problems with the catalog and directory after running DSNTIJTC. We used the following:

- Ran DSNTIJCX to ensure the integrity of the catalog indexes. The first step produced a return code of 4 as a result of no indexes being found for table space DSNDB06.SYSALTER (these objects will be created during the enabling of New Function Mode). The remaining steps produced a return code of zero.

Indexes can be put into advisory rebuild pending start during migration to DB2 Version 9 when columns are added to the index; DSNTIJRI rebuilds such indexes, and **Migration Step 20** deals with this. DSNTIJRI was executed successfully and we received a return code of 4, the result of several empty indexes.

In **Migration Step 21**, DSNTIJTM was executed to assemble, link-edit, bind, and invoke DSNTIAD. DSNTIJTM ran successfully.

In **Migration Step 22** we ran job DSNTIJSG according to the instructions specified. This step ended as expected.

Special considerations for Migration Step 22:

- In migration mode, job DSNTIJSG does not create any of the objects that are required for XML schema support. You can create these objects only after you have fully migrated to Version 9.
- If you bound special SPUFI packages and plans in Version 8, you need to bind those packages again in Version 9.1. You do not need to bind the plan again. For example, to update special SPUFI packages that were created for use by SPUFI users who require a TSO terminal CCSID of 1047, issue the following commands:

```
BIND PACKAGE(TIAP1047) MEMBER(DSNTIAP) -
     ACTION(REPLACE) ISOLATION(CS) ENCODING(1047) -
     LIBRARY('prefix.SDSNDBRM')
BIND PACKAGE(SPCS1047) MEMBER(DSNESM68) -
     ACTION(REPLACE) ISOLATION(CS) ENCODING(1047) -
     LIBRARY('prefix.SDSNDBRM')
BIND PACKAGE(SPRR1047) MEMBER(DSNESM68) -
     ACTION(REPLACE) ISOLATION(RR) ENCODING(1047) -
     LIBRARY('prefix.SDSNDBRM')
```

- In Version 9.1, SPUFI provides an option to select data with a cursor isolation level of Uncommitted Read. To add a special package and plan with ISO(UR) for SPUFI users who require a TSO terminal of CCSID 1047, issue the following commands:

```
BIND PACKAGE(SPUR1047) MEMBER(DSNESM68) -
     ACTION(REPLACE) ISOLATION(UR) ENCODING(1047) -
     LIBRARY('prefix.SDSNDBRM')
BIND PLAN(SPUR1047) -
     PKLIST(*.SPUR1047.DSNESM68, -
            *.TIAP1047.DSNTIAP) -
     ISOLATION(UR) ENCODING(1047) ACTION(REPLACE)
```

Because some views might have been marked with view regeneration errors during the migration to Version 9 compatibility mode, we performed **Migration Step 23** and identified the views with the following query:

```
SELECT CREATOR,NAME FROM SYSIBM.SYSTABLES
    WHERE TYPE='V' AND STATUS='R' AND TABLESTATUS='V';
```

The query found zero rows. However, if views had been found to have regeneration errors, the following alter command would correct the errors:

```
ALTER VIEW view_name REGENERATE;
```

In **Migration Step 24** we took another image copy of the directory and catalog after they were successfully migrated to V9, and submitted job DSNTIJIC (see **Migration Step 5** on page 81 for details). Execution of DSNTIJIC completed successfully.

The next step verifies the DB2 Version 9 subsystem that is now in Compatibility Mode; only selected Version 8 IVP jobs can be executed as outlined in the DB2 Version 9.1 for z/OS Installation Guide, **Migration Step 25**. After performing the necessary modifications, we ran these IVPs and received the expected results.

Finally, optional **Migration Step 26** deals with enabling WLM stored procedures by either executing the installation CLIST in MIGRATE mode or by editing and executing DSNTIJUZ. Additional information on enabling stored procedures is available in the *DB2 Installation Guide, (GC18-9846-00)*, under Chapter 10 page 361 *"Enabling stored procedures and user defined functions"*. Since we had already enabled WLM stored procedures under DB2 Version 8, this step was skipped.

DB2 V8 and V9 coexistence issues

We allowed the data sharing group to run in coexistence mode for several days while we tested various workloads and products for coexistence issues.

It is recommended that a data sharing group remain in coexistence mode for as brief a time period as necessary.

During this period we did not experience any problems.

Migrating the remaining members to compatibility mode

The next member to migrate in the data sharing group to DB2 Version 9 compatibility mode was DBS2. For us, this was a fairly simple process, which entailed the following steps:

1. Executing the installation CLIST
2. Executing the resultant DSNTIJUZ job
3. Replacing the Version 8 startup procs for the member being upgraded with their Version 9 equivalents. This is performed by executing DSNTIJMV step DSNTIPM
4. Starting the member.

So, beginning with the installation CLIST, we ran DSNTINST from the ISPF Command Shell (ISPF option 6) by entering the following command:

```
ex 'DB2.DB2910.SDSNCLST(DSNTINST)'
```

We filled in the first panel as shown in Figure 22 on page 86.

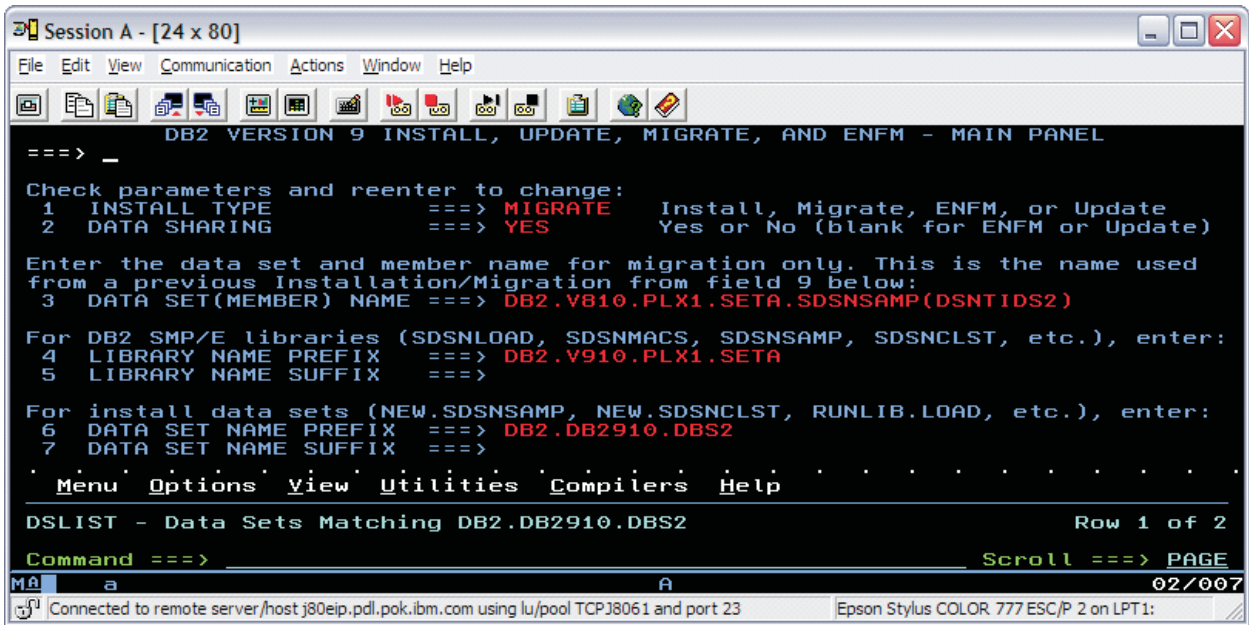


Figure 22. Executing DSNTINST in preparation for migrating the next member of the data sharing group

Pressing enter, we obtained the following pop-up screen as shown in Figure 23.

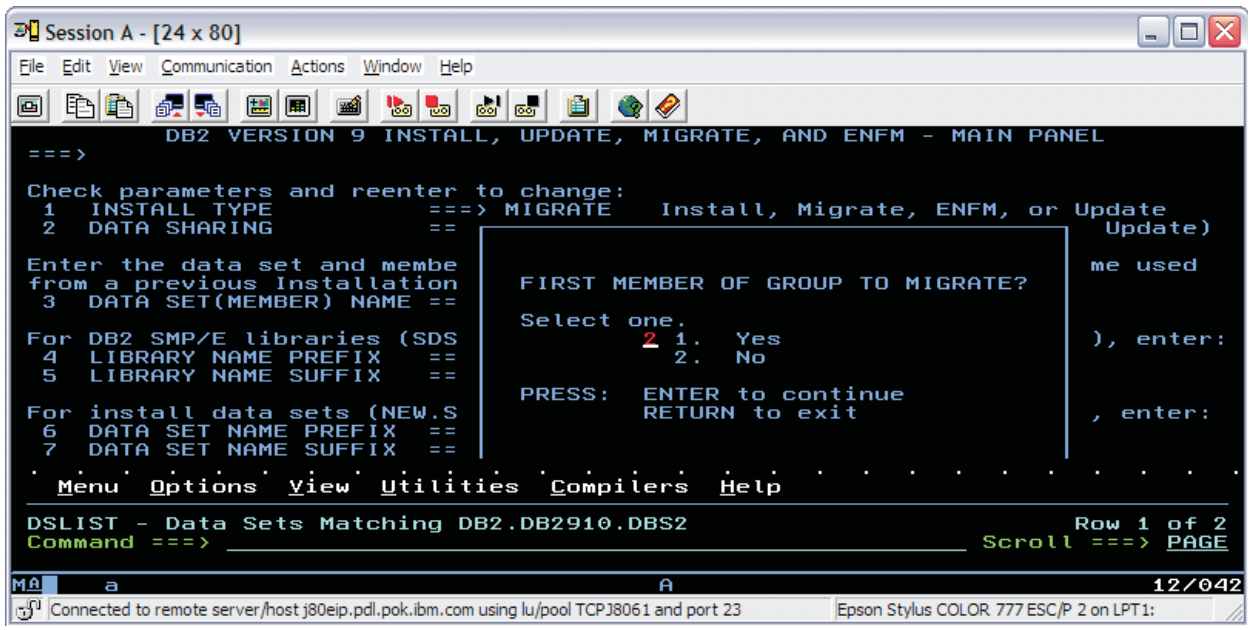


Figure 23. DSNTIPP2 pop-up screen

From this point, we scrolled through the panels and accepted the existing values with the exception of the name of the sample library on panel DSNTIPT. We maintain a separate sample library for each member of the data sharing group, so this field was updated accordingly to reflect DBS2, as shown in Figure 24 on page 87.

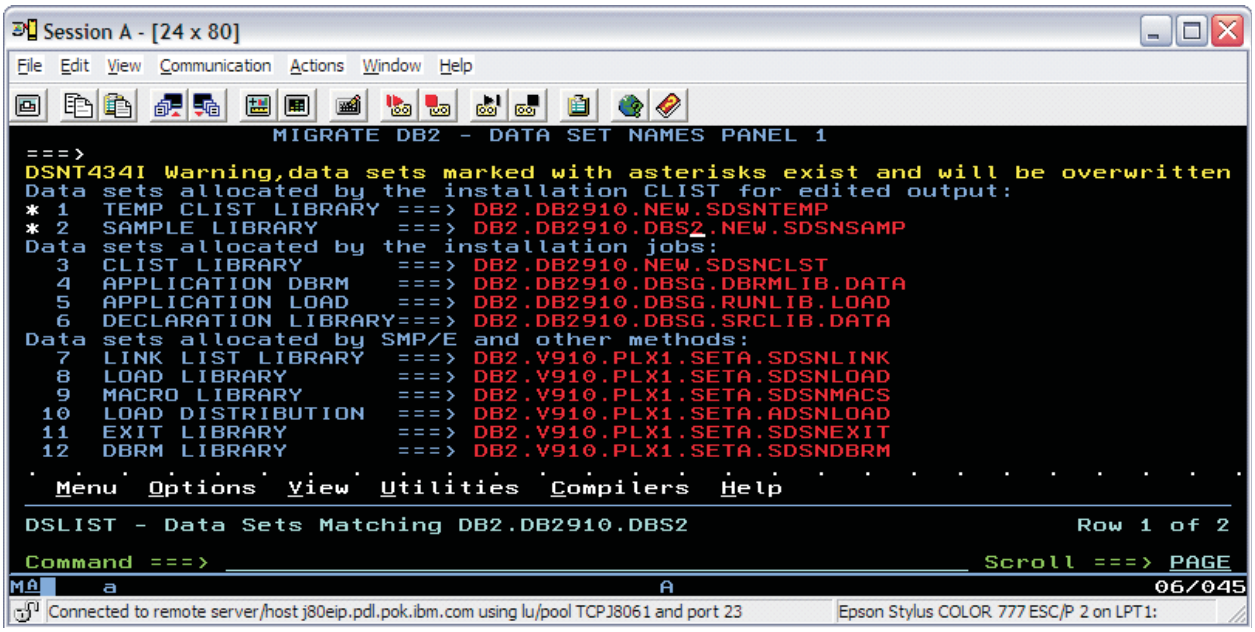


Figure 24. DSNTIPT - Data Set Names Panel 1

We placed the tailored migration JCL in DB2.DB2910.DBS2.NEW.SDSNSAMP as can be seen in the following example:

```
DSNT478I BEGINNING EDITED DATA SET OUTPUT
DSNT489I CLIST EDITING 'DB2.DB2910.DBS2.NEW.SDSNSAMP(DSNTIJMV)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS2.NEW.SDSNSAMP(DSNTIJTM)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS2.NEW.SDSNSAMP(DSNTIJGF)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS2.NEW.SDSNSAMP(DSNTIJFT)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS2.NEW.SDSNSAMP(DSNTIJFV)', FALL BACK JCL
DSNT489I CLIST EDITING 'DB2.DB2910.DBS2.NEW.SDSNSAMP(DSNTIJUZ)', INSTALL JCL
```

DBS2 was then brought down and DSNTIJUZ was executed after removing the SMP/E step; it ran successfully.

Next, we used steps RENAME and DSNTIPM of job DSNTIJMV to rename the existing Version 8 startup procedures for DBS2 and to add the new Version 8 startup procedures to proclib.

We then started DBS2 successfully in compatibility mode, as can be seen in the following example:

```
RESPONSE=J80
 DSN7100I  @DBS1 DSN7GCMD
 *** BEGIN DISPLAY OF GROUP(DSNDBSG ) GROUP LEVEL(910) MODE(C)
                   PROTOCOL LEVEL(2)  GROUP ATTACH NAME(DBSG)
 ------------------------------------------------------------------------
 DB2                                      DB2 SYSTEM    IRLM
 MEMBER   ID  SUBSYS CMDPREF   STATUS     LVL NAME      SUBSYS IRLMPROC
 -------- --- ---- --------   --------   --- --------  ----   --------
 DBS1       1 DBS1   @DBS1     ACTIVE     910 J80       IRS1   DBS1IRLM
 DBS2       2 DBS2   @DBS2     ACTIVE     910 JB0       IRS2   DBS2IRLM
 DBS3       4 DBS3   @DBS3     ACTIVE     810 JF0       IRS3   DBS3IRLM
 ------------------------------------------------------------------------
```

We followed the same process for the remaining member of the data sharing group, resulting in all members being in compatibility mode as shown below:

```
              RESPONSE=J80
DSN7100I  @DBS1 DSN7GCMD
*** BEGIN DISPLAY OF GROUP(DSNDBSG ) GROUP LEVEL(910) MODE(C)
                  PROTOCOL LEVEL(2)  GROUP ATTACH NAME(DBSG)
------------------------------------------------------------------------
DB2                                     DB2 SYSTEM    IRLM
MEMBER   ID  SUBSYS CMDPREF   STATUS    LVL NAME      SUBSYS IRLMPROC
-------- --- ----   --------  --------  --- --------  ----   --------
DBS1       1 DBS1   @DBS1     ACTIVE    910 J80       IRS1   DBS1IRLM
DBS2       2 DBS2   @DBS2     ACTIVE    910 JB0       IRS2   DBS2IRLM
DBS3       4 DBS3   @DBS3     ACTIVE    910 JF0       IRS3   DBS3IRLM
------------------------------------------------------------------------
```

Migrating to new function mode

After we migrated all members of the data sharing group to compatibility mode, we had to convert the DB2 catalog to exploit the new functions introduced by DB2 Version 9. The process is outlined below:

- "Preparing for new function mode"
- "Enabling new function mode" on page 91
- "Running in new function mode" on page 93
- "Verifying the installation using the sample applications" on page 93

Preparing for new function mode

Before enabling-new-function mode, ensure that the following steps are taken:

- **Important:** All members of a data sharing group must have been successfully migrated to Version 9.1 compatibility mode before commencing the enabling-new-function mode process.
- A point of consistency needs to be created for the catalog and directory before enabling new-function mode. The Quiesce Utility should be used to establish a point of consistency for the catalog and directory table spaces; note that DSNDB01.SYSUTILX should be quiesced by itself. Updates to the DB2 catalog and directory should be avoided while in enabling new-function mode. **This is the only time when applications were brought down; planning is therefore essential to reduce the amount of down time.**
- Run the installation CLIST using the ENFM option on panel DSNTIPA1.

After insuring a point of consistency for the catalog and directory , the installation CLIST was executed; panel DSNTIPA1 was completed as shown in Figure 25 on page 89.

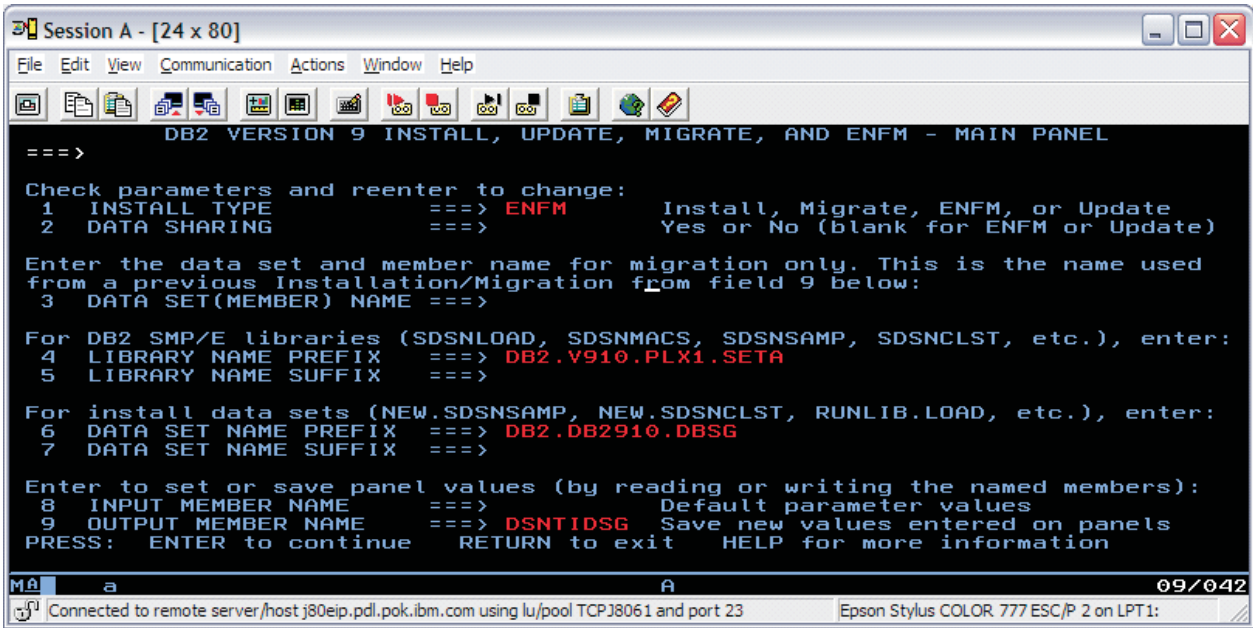


Figure 25. Executing DSNTINST in preparation for enabling-new-function-mode

Press enter twice to display panel DSNTIP00; space was calculated as shown in Figure 26 and Figure 27 on page 90.

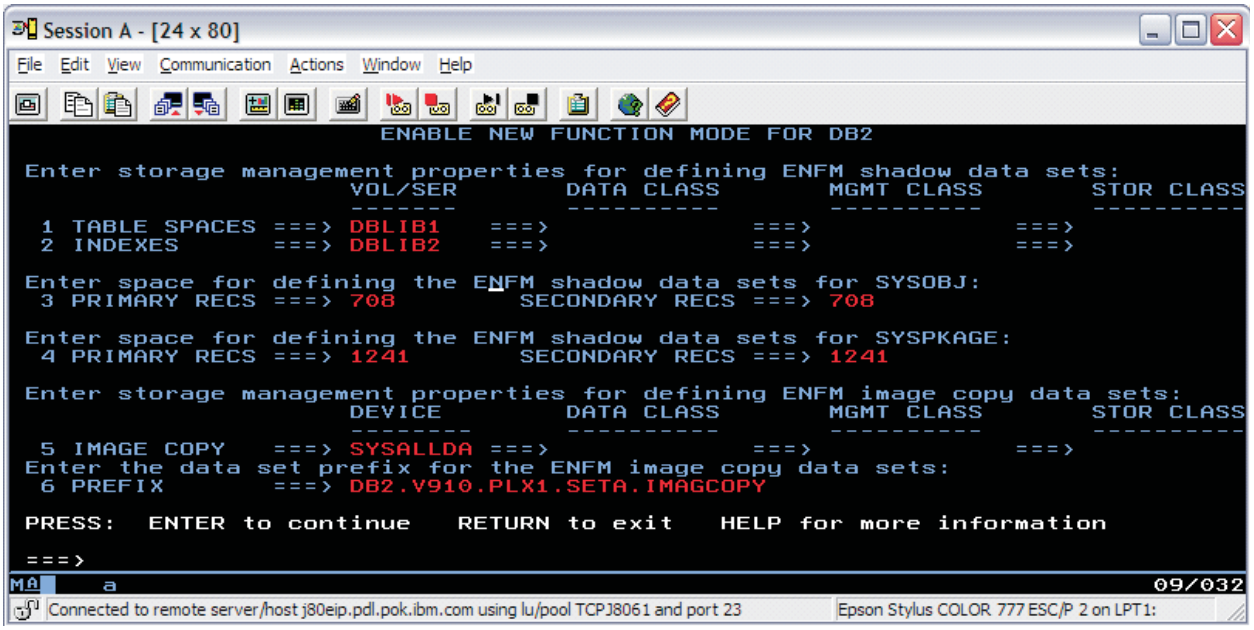


Figure 26. DSNTIP00 first panel

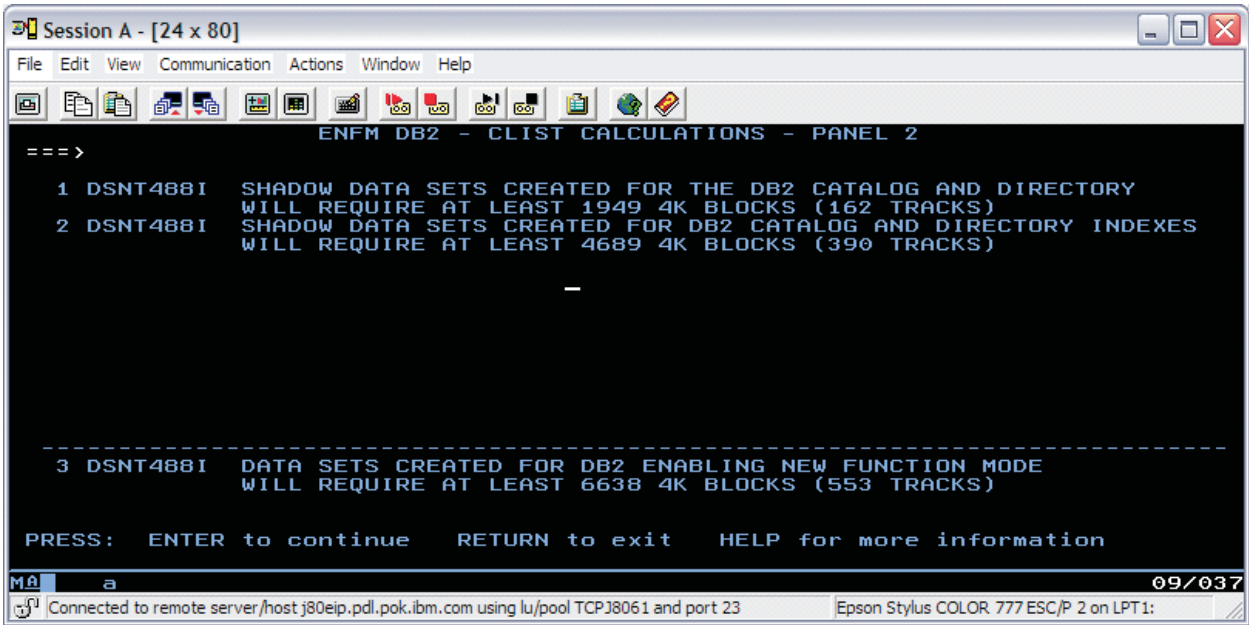


Figure 27. DSNTIP00 second panel

We accepted calculated values and pressed enter to continue.

This was the last panel displayed. When we pressed enter, the generation of the enabling-new-function mode job along with the DB2 Version 9 sample jobs, occurred as shown in the following three screen images:

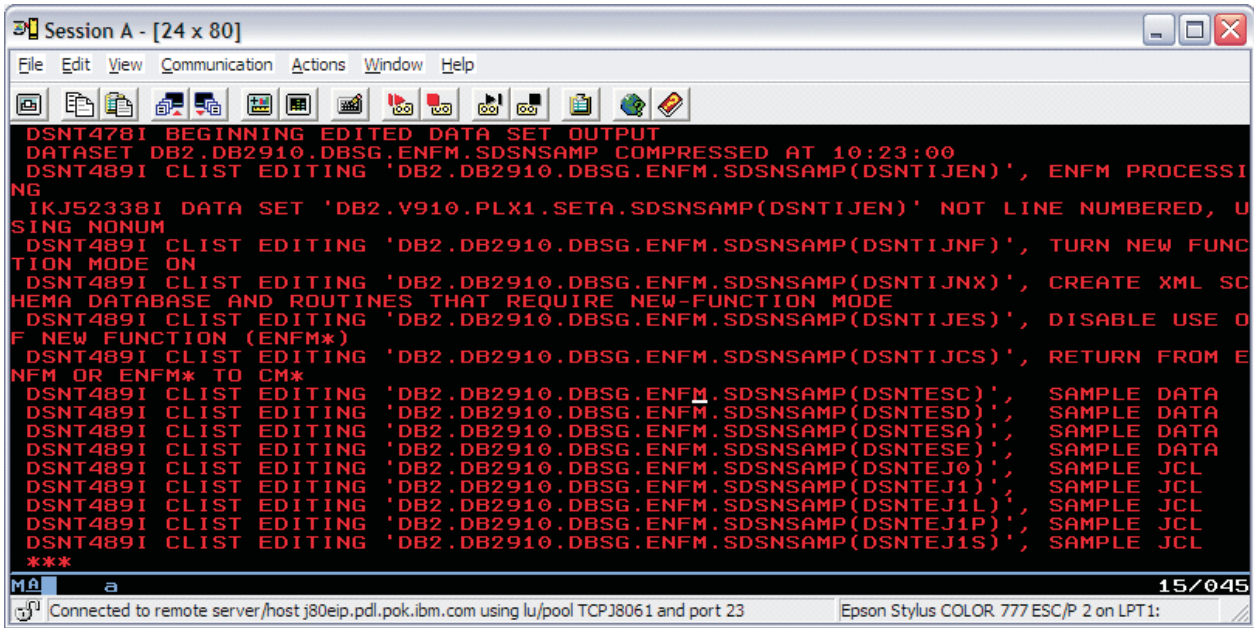


Figure 28. DSNT478I beginning data set output

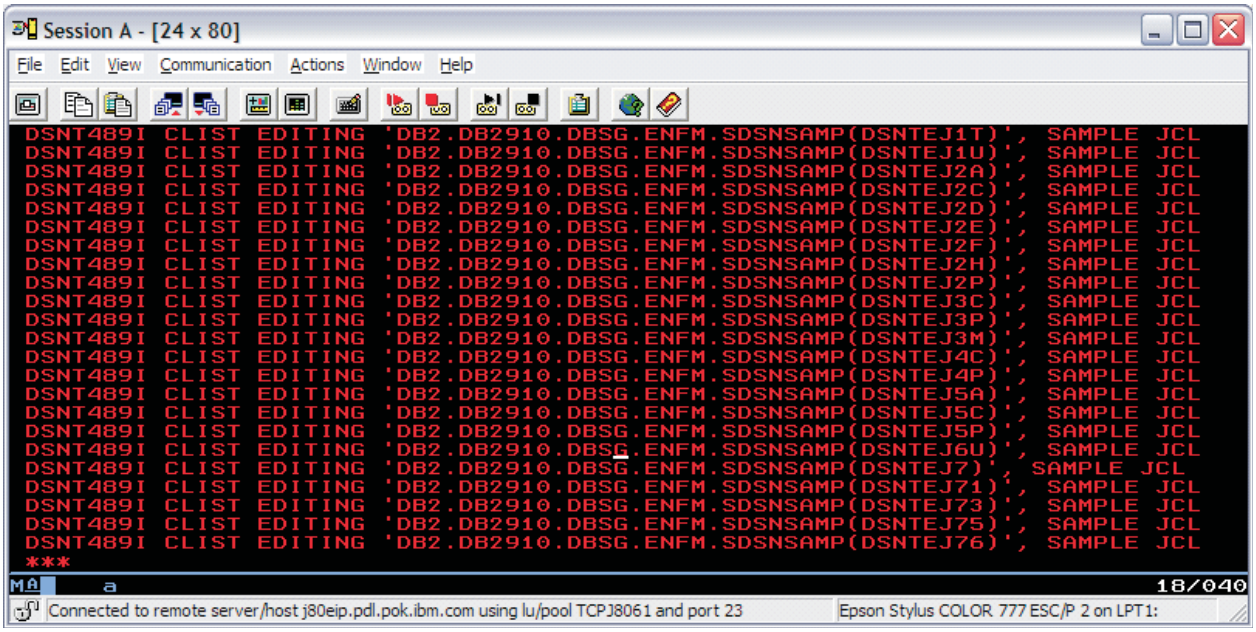


Figure 29. DSNT489I CLIST editing

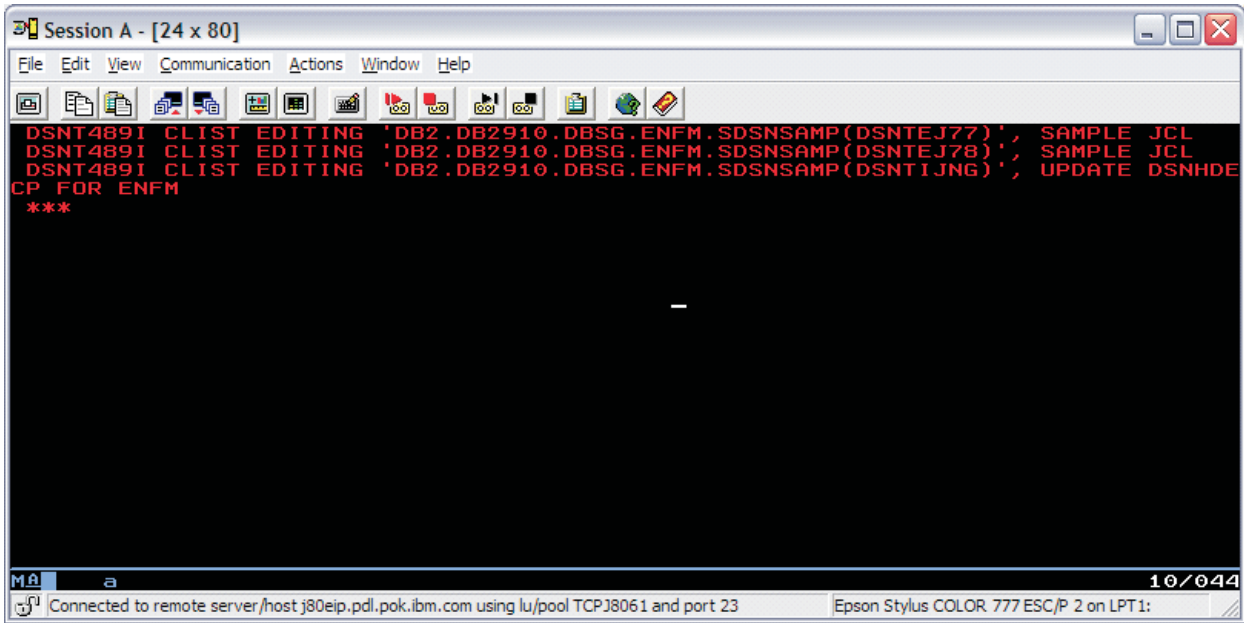


Figure 30. Completion of the preparation before enabling Version 9 new function mode

Enabling new function mode

For **Step 1**, we executed DSNTIJEN and received the messages shown in Figure 27; DSNTIJEN performs the following functions:

- Saves the current RBA or LRSN in the BSDS
- Changes types and lengths of existing catalog columns
- Changes buffer pool for the SYSOBJ table space
- Changes page size of the SYSOBJ table space

- Copies the RTS from the user table spaces to new table spaces in the catalog
- Creates a new index, DSNRTX03, on SYSINDEXSPACESTATS.

The following are the results of the conversion steps:

```
DSNUECM0 - CATENFM START PHASE 1 STARTED
DSNUECM0 - CATENFM START STATUS - VERIFYING CATALOG IS AT CORRECT LEVEL FOR ENFM
DSNUECM0 - CATENFM START PHASE 1 COMPLETED
DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = DSNENFM.ENFM0100
DSNUGTIS - PROCESSING SYSIN AS EBCDIC
DSNUGUTC -  CATENFM CONVERT INPUT SYSOBJ
DSNUECM0 - CATENFM CONVERT PHASE 1 STARTED
DSNUECM0 - CATENFM CONVERT PHASE 1 COMPLETED
DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0


DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = DSNENFM.ENFM0110
DSNUGTIS - PROCESSING SYSIN AS EBCDIC
DSNUGUTC -  CATENFM CONVERT INPUT SYSPKAGE
DSNUECM0 - CATENFM CONVERT PHASE 1 STARTED
DSNUECM0 - CATENFM CONVERT PHASE 1 COMPLETED
ALTER COLUMN "SEQNO" SET DATA TYPE
DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0



DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = DSNENFM.ENFM1200
DSNUGTIS - PROCESSING SYSIN AS EBCDIC
DSNUGUTC -  CATENFM CONVERT INPUT SYSRTSTS
DSNUECM0 - CATENFM CONVERT PHASE 1 STARTED
DSNUECM0 - CATENFM CONVERT PHASE 1 COMPLETED
DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
```

Step 2 recommends taking an image copy of the catalog and directory at this point.

For **Step 3**, we ran DSNTIJNF which places the DB2 subsystem in new function mode; the job ended with return code zero as shown below:

```
DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = DSNENFM.ENFM9700
DSNUGTIS - PROCESSING SYSIN AS EBCDIC
DSNUGUTC -  CATENFM COMPLETE
DSNUECM0 - CATENFM COMPLETE PHASE 1 STARTED
DSNUECM0 - CATENFM COMPLETE STATUS - ENTERING NEW FUNCTION MODE (NFM).
DSNUECM0 - CATENFM COMPLETE PHASE 1 COMPLETED
DSNUGBAC - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
```

Step 4 is concerned with executing DSNTIJNX, which creates objects for XML Schema Repository (XSR) support. We submitted this job and it ran to completion successfully.

In **Step 5**, DSNTIJNG rebuilds DSNHDECP to specify new function mode as the default by specifying NEWFUN=YES.

Note: If you use more than one DSNHDECP member, modify and update each to use NEWFUN=YES.

To verify the data sharing group was now in new function mode, we issued a DISPLAY GROUP COMMAND; as can be seen in Figure 31 on page 93, MODE(N) has replaced MODE(C):

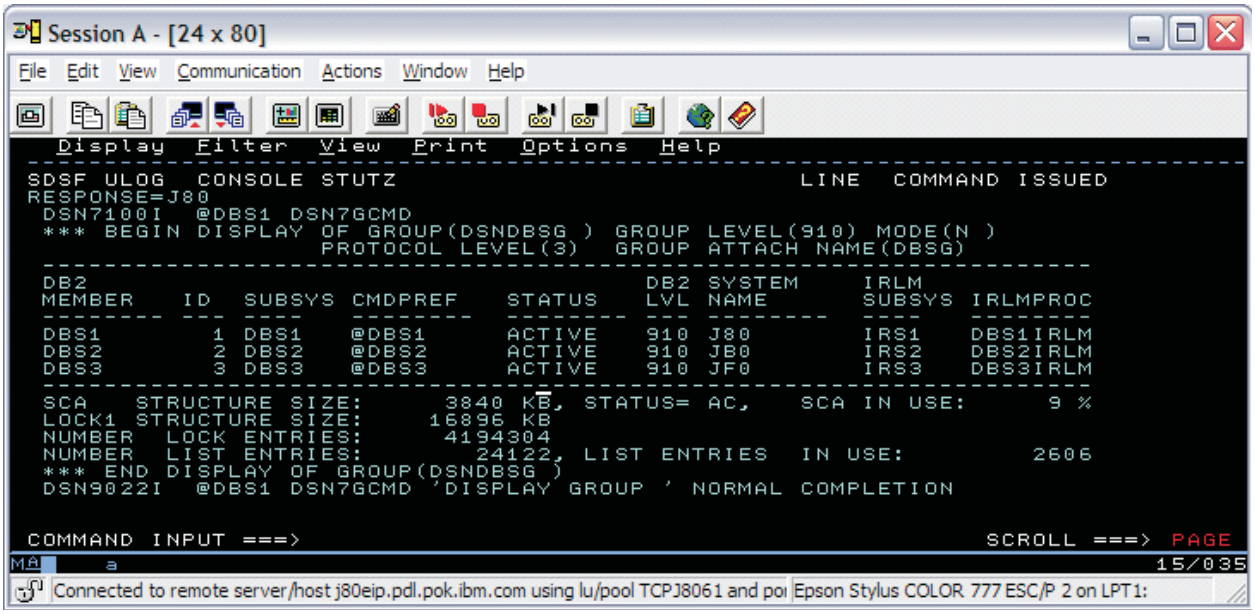


Figure 31. DISPLAY GROUP command showing the data sharing group is now in new function mode

Running in new function mode

Once in new function mode, it is recommended to alter any frequently accessed buffer pools so that their pages are fixed in real storage, thereby avoiding the overhead involved for DB2 to fix and free pages each time an I/O operation is performed. For I/O intensive workloads, this processing time can amount to as much as 10%. To fix pages in storage, the PGFIX parameter of the ALTER BPOOL command is used as shown below:

```
ALTER BPOOL(buffer_pool_name) VPSIZE(virtual_page_size) PGFIX(YES)
```

Note that you should verify that sufficient real storage is available for fixing buffer pool pages before issuing the ALTER BPOOL command.

Verifying the installation using the sample applications

Using the sample applications provided in DB2.DB2910.DBSG.ENFM.SDSNSAMP, we performed verification of DBSG migration to DB2 Version 9 as outlined below. Note that of the seven verification phases available, we ran only those phases and their associated jobs that applied to our specific environment.

Phase 0 is comprised of a single job, DSNTEJ0, that is used to free all objects that were created by running any of the seven verification phases. This permits the verification phases to be executed again in their entirety without the possibility of failure as a result of objects having been previously created.

Phases 1 through **3** are used to test the TSO and batch environments, including user-defined functions.

Phase 4 addresses IMS.

Phase 5 addresses CICS.

Phase 6 initializes sample tables and stored procedures for distributed processing.

Finally, **Phase 7** is used for the testing of DB2's Large Object feature (LOB) using sample tables, data, and programs.

We added the following JCLLIB statement after the JOB statement for all verification jobs that were executed:

```
//         JCLLIB ORDER=DB2.DB2910.DBSG.PROCLIB
```

Recall that in **Migration Step 13** job DSNTIJMV was executed to add catalogued procedures to proclib; however, rather than directing the output of this step to SYS1.PROCLIB, it was directed to the newly created data set DB2.DB2910.DBSG.PROCLIB. This library must be APF authorized (we dynamically added it to the APF authorization list before proceeding).

Planning for verification

Before performing any of the verification phases, you must make certain decisions about your verification strategy. DB2 system administrators and system administrators for ISPF, TSO, batch, IMS, and CICS must be involved in these decisions. Working together, these system administrators do the following:

- Determine the verification phases that you plan to perform. Examine the description of each verification phase in this chapter, and determine which phases apply to your needs.
- Identify any phases that you want to modify before you perform them. Verification is designed to run with little interaction on your part. This chapter does not discuss how to modify any of the phases, but you can adapt any of the seven phases to your needs. If this is your intent, identify and describe any modifications you plan to make.
- Establish additional testing steps to complete the verification. The verification phases and the jobs that you run to perform them are valuable tools for testing DB2. They are not a substitute for a thorough subsystem test. You must plan and perform your own additional testing to complete the verification. To help you assess which additional tests might be necessary, examine the sample applications that are provided with DB2.

We executed the following IVP jobs after every change to the environment (hardware, software).

DSNTEJ1
DSNTEJ1L
DSNTEJ1P
DSNTEJ1S
DSNTEJ2A
DSNTEJ2C
DSNTEJ2D
DSNTEJ2E
DSNTEJ2H
DSNTEJ2P
DSNTEJ3C
DSNTEJ3M
DSNTEJ3P
DSNTEJ6U
DSNTEJ7
DSNTEJ71
DSNTEJ73
DSNTEJ75
DSNTEJ76
DSNTEJ77
DSNTEJ78

You may choose to execute more or less than what we schedule to run. Based on your needs, you may choose to run the IVPs on a different cycle than what we have set up.

Chapter 8. Using z/OS UNIX System Services

The following topics describe our experiences with z/OS UNIX System Services (z/OS UNIX):

z/OS UNIX enhancements in z/OS V1R9

The following topics describe our test experiences with z/OS UNIX enhancements in z/OS V1R9:

AUTOMOVE consistency

z/OS UNIX improved the way AUTOMOVE settings affect file system movements due to situations such as system failures and shutdowns. Prior to z/OS V1R9, the AUTOMOVE specification was not honored if a file system was mounted in a mode for which the physical file system (PFS) provides *sysplex-aware* capability. The movement of file systems is now more consistent and predictable in these situations.

The levels of sysplex awareness for a PFS are:

sysplex-unaware
Refers to the PFS capability (restriction) that a file system can only be mounted locally on one system in the shared file system configuration. All other systems obtain access to the file system by function shipping operations to the file system server. For example, the HFS PFS is sysplex-unaware for RDWR mounts.

sysplex-aware
Refers to the PFS capability to locally mount a file system on all systems in the shared file system configuration for a particular mount mode, READ or RDWR. For example, the HFS PFS and zFS PFS are sysplex-aware for READ mounts. The zFS PFS can be configured to be sysplex-aware or sysplex-unaware in RDWR mode.

fully sysplex-aware
Refers to the PFS capability to locally mount a file system on all systems in the shared file system configuration for either READ or RDWR mount modes. This is the intended design direction for zFS.

We tested various scenarios that included soft shutdowns, system failures, and PFS termination (zFS). For each test, there were file systems defined to use different AUTOMOVE settings. Our Parallel Sysplex consists of nine active members. The resulting behavior of the settings we tested was exactly as expected. The file

systems that had the AUTOMOVE(YES) attribute had their ownership moved to another system in the sysplex. File systems that had the AUTOMOVE(NO) attribute stayed known to z/OS UNIX; however, they were not accessible since the owning system or PFS was no longer accessible. File systems that were mounted with the UNMOUNT attribute were unmounted, as expected, when the owning system was shutdown or failed.

The final disposition of a file system in the various recovery or shutdown functions is dependent on:

- The AUTOMOVE attribute, as specified at MOUNT time
- The capabilities of the PFS:
  - Whether or not the PFS supports recovering the file system at all on another system. A PFS such as TFS (where the data resides in virtual storage) cannot support moving a file system from one system to another in the various recovery scenarios. This is commonly referred to as a *never move* PFS.
  - Whether or not the PFS provides sysplex-aware capability for the mount mode (RDWR or READ)—thus indicating whether or not the file system is locally mounted.
- Automount managed file systems, which have their own set of behaviors

The above attributes influence the various file systems' recovery and shutdown processing for:

- Member gone recovery (also known as dead system recovery)
- PFS termination
- z/OS UNIX shutdown
- File system shutdown (such as MODIFY BPXOINIT,SHUTDOWN=FILESYS)
- Multiple file system move processing (such as SETOMVS FILESYS,FROMSYS=)

The logic to convert SYSLIST to AUTOMOVE was removed in z/OS V1R9; SYSLIST is now honored for both sysplex-aware and sysplex-unaware file systems. Also, NOAUTOMOVE is now honored for both sysplex-aware and sysplex-unaware file systems.

Unmount of automount file systems

Another change in z/OS UNIX that we tested was the automount facility. Automount was changed to have the UNMOUNT attribute if the parent file system also has the UNMOUNT attribute. Also, automount will not inherit any other AUTOMOVE attributes, such as AUTOMOVE(NO) or SYSLIST, since these are not supported for sysplex-aware file systems. Automount is a sysplex-aware file system. Automount will now do a **getmntent** call for the parent directory to check its AUTOMOVE attributes. If AUTOMOVE(UNMOUNT) is found, it will set that attribute on its **mount** call.

For example, if you set up automount to manage a directory on top of /etc, which has the UNMOUNT attribute, then when the system shuts down or fails for some reason, the automount-managed file system will also be unmounted.

SMF record type 92 subtype 14 for z/OS file deletion and rename

SMF record type 92 reports activity of mounted file systems and files. In z/OS V1R9, z/OS UNIX SMF 92 records were enhanced with a new record subtype 14 to report when a file or directory is deleted or renamed. Information will be collected

in a SMF 92 Subtype 14 record. The installation must set up monitoring for SMF type 92 subtype 14 records in order to collect this information. In a shared file system environment, recording occurs on the user's system where the command was issued.

Some of the operations for which an SMF type 92 subtype 14 record will be written are: **vn_remove** and **v_remove**, **vn_rename** and **v_rename**, and **vn_rmdir** and **v_rmdir**.

For a description of the SMF type 92 subtype 14 record format, see *z/OS MVS System Management Facilities (SMF)*.

z/OS UNIX couple data set BPXOINIT and XCF DISPLAY and message consistency

In z/OS V1R9, the IXC358I message returned by the command DISPLAY XCF,COUPLE,TYPE=BPXMCDS now displays the values for mount entries and automount rules (AMTRULES), as defined in the couple data set (CDS). It also displays the CDS version. The command MODIFY BPXOINIT,FILESYS=DISPLAY will continue to display the values for mounts, automount rules (AMTRULES), and CDS version.

Distributed Byte Range Lock Manager (BRLM) is the only supported byte range locking method since z/OS V1R6. Since the z/OS UNIX couple data set is enabled for distributed BRLM by default, the BRLM references were removed from the F BPXOINIT,FILESYS=DISPLAY display and from some messages. The message BPXF041I was replaced with message BPXF242I, eliminating BRLM information.

The following are display examples. Note the new message ID of BPXF242I for the F BPXOINIT,FILESYS=D display and the elimination of BRLM information. Also, note that message IXC358I, displayed by the D XCF,COUPLE,TYPE=BPXMCDS command, now includes FORMAT DATA for VERSION, MOUNTS, and AMTRULES, under ADDITIONAL INFORMATION.

```
-F BPXOINIT,FILESYS=D
 BPXM027I COMMAND ACCEPTED.
 BPXF242I 2007/06/22 11.46.03 MODIFY BPXOINIT,FILESYS=DISPLAY,GLOBAL
 SYSTEM   LFS VERSION ---STATUS--------------------- RECOMMENDED ACTION
 J80        1.  9.  0 VERIFIED                      NONE
 JF0        1.  8.  0 VERIFIED                      NONE
 JC0        1.  9.  0 VERIFIED                      NONE
 TPN        1.  9.  0 VERIFIED                      NONE
 JB0        1.  9.  0 VERIFIED                      NONE
 J90        1.  8.  0 VERIFIED                      NONE
 JE0        1.  9.  0 VERIFIED                      NONE
 JA0        1.  9.  0 VERIFIED                      NONE
 Z0         1.  9.  0 VERIFIED                      NONE
 CDS VERSION=  2        MIN LFS VERSION=  1.  8.  0
 DEVICE NUMBER OF LAST MOUNT=    8506
 MAXIMUM MOUNT ENTRIES=    1100   MOUNT ENTRIES IN USE=     773
 MAXIMUM AMTRULES=           51   AMTRULES IN USE=            9
MAXSYSTEM=                  16
BPXF040I MODIFY BPXOINIT,FILESYS PROCESSING IS COMPLETE.
```

```
-D XCF,COUPLE,TYPE=BPXMCDS
 IXC358I  11.47.40  DISPLAY XCF 936
 BPXMCDS COUPLE DATA SETS
 PRIMARY    DSN: SYS1.OMVS.CDS10
            VOLSER: CDSOMP     DEVN: 2423
            FORMAT TOD         MAXSYSTEM
```

```
            03/20/2007 13:39:27       16
            ADDITIONAL INFORMATION:
              FORMAT DATA
               VERSION(2)
               MOUNTS(1100)   AMTRULES(51)
 ALTERNATE  DSN: SYS1.OMVS.CDS11
            VOLSER: COUPL4     DEVN: 461A
            FORMAT TOD         MAXSYSTEM
            08/23/2005 10:13:33       16
            ADDITIONAL INFORMATION:
             FORMAT DATA
              VERSION(2)
              MOUNTS(1100)   AMTRULES(51)
```

Message BPXI078I replaces message BPXI050I, eliminating BRLM references (THE VALUE OF DISTBRLM IS 1). This message is issued when the values of the new CDS are greater than the one it replaced. Notice that there are no BRLM references in the following example of message BPXI078I:

```
BPXI078I THE PRIMARY CDS SUPPORTS A LIMIT OF 1200 MOUNTS AND
 A LIMIT OF 52 AUTOMOUNT RULES.  THE CDS VERSION IS 2.
```

If XCF failed the ACOUPLE request, message IXC255I only described the mismatch in terms of internal record names, which did not help the system programmer to figure out how to format the alternate CDS. In z/OS V1R9, additional information for BPXMCDS will be added to message IXC255I to help in determining the error.

For example, if we try to add an alternate CDS (SYS1.OMVS.CDS10) that has values less than the primary CDS for MOUNTS and AMTRULES, we receive the following:

```
-SETXCF COUPLE,ACOUPLE=(SYS1.OMVS.CDS10),TYPE=BPXMCDS

 IXC309I SETXCF COUPLE,ACOUPLE REQUEST FOR BPXMCDS WAS ACCEPTED
 IXC260I ALTERNATE COUPLE DATA SET REQUEST FROM SYSTEM
 J80 FOR BPXMCDS IS NOW BEING PROCESSED.
 DATA SET:  SYS1.OMVS.CDS10
 IXC255I UNABLE TO USE DATA SET
 SYS1.OMVS.CDS10
 AS THE ALTERNATE FOR BPXMCDS:
 ALLOWABLE SIZE OF BPXFSMPT RECORDS IS LESS THAN CURRENT PRIMARY
 RELEVANT BPXMCDS COUPLE DATA SET FORMAT INFORMATION
 PRIMARY
   FORMAT LEVEL:    VERSION(2)
   FORMAT KEYWORDS: MOUNTS(1100)   AMTRULES(51)
 ALTERNATE
  FORMAT LEVEL:    VERSION(2)
  FORMAT KEYWORDS: MOUNTS(1099)   AMTRULES(50)
IXC255I UNABLE TO USE DATA SET
SYS1.OMVS.CDS10
AS THE ALTERNATE FOR BPXMCDS:
ALLOWABLE SIZE OF BPXFSAMT RECORDS IS LESS THAN CURRENT PRIMARY
RELEVANT BPXMCDS COUPLE DATA SET FORMAT INFORMATION
PRIMARY
  FORMAT LEVEL:    VERSION(2)
  FORMAT KEYWORDS: MOUNTS(1100)   AMTRULES(51)
ALTERNATE
  FORMAT LEVEL:    VERSION(2)
  FORMAT KEYWORDS: MOUNTS(1099)   AMTRULES(50)
IXC250I ALTERNATE COUPLE DATA SET REQUEST FAILED FOR DATA SET
SYS1.OMVS.CDS10 FOR BPXMCDS:
CONSISTENCY CHECKING FAILED FOR THE NEW ALTERNATE DATA SET
```

z/OS UNIX tools — fsdiruse sample

It's almost inevitable: File systems fill up. Determining what is using up the most space is often a painful hunt through the file system. When it happens unexpectedly, panic can set in. Complicating the issue, many of our file systems these days have extensive directory structures, have other file systems mounted underneath, or have multiple products, users, or applications using them (such as /tmp and /etc).

As an aid to this perpetual problem, we modified one of the sample programs from the z/OS UNIX tools Web site to help provide us with a means to determine where the usage is within a file system's directories.

Our **fsdiruse** program does the following:

- Displays summary usage of a file system (bytes used, number of files and subdirectories)
- Breakdown of the usage in the first level subdirectories.
- Reports only for the target file system (skips symbolic links and mount points)

Figure 32 shows a sample of the **fsdiruse** output for /Z1/tmp:

```
Bytes used    Dirs      Files       Sub-directory
---------- ---------- ---------- ------------------------------
         0          0          0  /Z1/tmp/IBMRAC
         0          1          0  /Z1/tmp/DB2GWLMJ
         0          1          0  /Z1/tmp/DBXGWLM
     93379          2         36  /Z1/tmp/wwwlogs
         0          0          0  /Z1/tmp/mqsi-CSQ1BRK_servlet_workdir
         0          1          0  /Z1/tmp/mkdir2tmpv1r52
         0          1          0  /Z1/tmp/mkdir2tmpv1r5
    837562          2         36  /Z1/tmp/iwl
         0          2          0  /Z1/tmp/fw
         0          0          1  /Z1/tmp/.cssm
    254325          0          2  /Z1/tmp/logarch
   2689972          6         26  /Z1/tmp/wasusr1
   1563366          0          4  /Z1/tmp/java
         0         14          0  /Z1/tmp/WQ1GWLMJ
         0          0          0  /Z1/tmp/bpxwh2z.WASADM1.temp.zfs
---------- ---------- ---------- ------------------------------
   5438604         45        105   in subdirectories
   3798637                   102   not in subdirectories
   9237241         46        207   total

block spc  char spc   ext links  pipes/fifo  symlinks   sockets
---------- ---------- ---------- ---------- ---------- ----------
         0          1          0         72          0          0
```

Figure 32. Sample output from our fsdiruse tool, run on the /Z1/tmp directory

From the output shown in Figure 32, we can quickly see that there are two subdirectories using most of the space (/Z1/tmp/wasusr1 and /Z1/tmp/java) and are good candidates to warrant a closer look.

Since /Z1/tmp/wasusr1 has six directories, **fsdiruse** can be run again with the /Z1/tmp/wasusr1 directory as its parameter to show the breakdown within those directories.

Downloading, compiling, and running fsdiruse

The source code for **fsdiruse** can be found and downloaded from the Examples section of our Web site at www.ibm.com/servers/eserver/zseries/zos/integtst/.

To compile, ftp this file to a directory of your choice and run **make fsdiruse**. Defaults for **make** should generally be sufficient to compile.

To run, enter:

```
fsdiruse directory
```

where *directory* is the target directory to scan.

The **fsdiruse** program will only report from the requested directory and lower. If you want to see the usage of the entire file system, you will need to point it to the mount point for that file system (for instance, /Z1/tmp versus /Z1/tmp/java).

We placed a copy of the code in the /usrbin directory on both of our sysplexes. Users simply need to add /usrbin to their PATH environment variable to use it (along with other tools) or they can point to it directly.

z/OS UNIX tools

Our **fsdiruse** tool is modified from the **dirsize** tool, available from the z/OS UNIX Tools and toys Web site at www.ibm.com/servers/eserver/zseries/zos/unix/bpxa1toy.html. This site contains many sample programs that you might find helpful, either "as is" or, as we did, modify them to suit our needs. Many can be added to your own toolbox!

Using the _UNIX03 environment variable in the z/OS UNIX shell

The UNIX 03 Product Standard is the mark for systems conforming to Version 3 of the Single UNIX Specification (SUS V3). It is a significantly enhanced version of the UNIX 98 Product Standard. For more information on this standard, go to The Open Group Web site (www.unix.org).

In z/OS V1R8, some z/OS UNIX utilities implemented support for the UNIX 03 specification. _UNIX03 is an environment variable. When _UNIX03 is set to `YES`, the utilities that have implemented support for the UNIX 03 specification will conform to it. Note that this variable is only needed when the syntax or behavior of the new implementation conforming to UNIX 03 conflicts with the existing implementation.

The following are two utilities that support the UNIX 03 specification:

- **cp**
- **mv**

cp utility

In z/OS V1R8, the OMVS shell utility **cp** has three new options (**-H**, **-L**, **-P**) to handle symbolic link processing during a recursive copy (**-R** or **-r** option flags). However, there was already an existing **-P** option for the **cp** utility. It was used for specifying the parameters needed to create a sequential data set. To resolve this conflict, use the _UNIX03 environment variable to specify whether **cp** is to process **-P** for symbolic links handling or **-P** for sequential data set creation. If _UNIX03 is set to `YES`, **cp** will process **-P** for symbolic links handling. If _UNIX03 is set to anything other than `YES`, **cp** will process **-P** for creating a sequential data set.

Another new option for the **cp** utility is **-W**. It works the same way as today's **-P** option. It is provided so that you can create sequential data sets while the _UNIX03 environment variable is set to YES as well.

Here are what the three new **cp** options do:

-H When the **-H** option is specified, **cp** follows symbolic links specified as a source operand on the command line. Following a symbolic link means that an exact copy of the file that is linked will be created rather than a copy of the symbolic link itself.

-L When the **-L** option is specified, **cp** behaves the same way it does when **-H** is specified. However, it also follows the symbolic links that are found during tree traversal.

-P When the **-P** option is specified, **cp** does not follow any symbolic links.

Another new option for the cp utility is:

-W

-W works the same way as today's **-P** option. It is provided so that users can create sequential data sets while the _UNIX03 environment variable is set to YES.

Examples of z/OS UNIX utilities that implement support for the UNIX 03 specification

Set the _UNIX03 environment variable to YES.

```
export _UNIX03=YES
```

Recursively copy directory dir1 to dir2. Use the **-P** option so that no symbolic links are followed.

```
cp –r –P dir1 dir2
```

Set the _UNIX03 environment variable to anything other than YES.

```
export _UNIX03=NO
```

Next, use the **-P** option to specify the parameters needed to create a sequential data set. The command below will copy file1 into a new sequential data set named uss.test0.

```
cp -P "RECFM=U,space=(5,1)" file1 "//'uss.test0'"
```

Leave the _UNIX03 option set to anything other than YES. Use the **-W** option to create a sequential data set called uss.test1.

```
cp -W "seqparms='RECFM=U,space=(5,1)'" file1 "//'uss.test1'"
```

Set the _UNIX03 environment variable to YES. Use the **-W** option to create a sequential data set called uss.test2. The **cp -W** command behaves the same regardless of the value of the _UNIX03 variable.

```
cp -W "seqparms='RECFM=U,space=(5,1)'" file1 "//'uss.test2'"
```

mv utility

In z/OS V1R8, the z/OS UNIX utility **mv** has a new option as well (**-W**). It serves the exact same purpose as the existing **mv** option **-P**. It is implemented purely for consistency between the **cp** and **mv** utilities. Since the **mv** utility does not have any option conflict issues, the _UNIX03 environment variable does not need to be set to YES for **mv** to process the **-W** option.

z/OS zFS enhancements in z/OS V1R9

The following topics describe our experiences with the Distributed File Service zFS enhancements in z/OS V1R9.

- "zFS format authorization"
- "New aggregate full message from zFS"

zFS format authorization

It is a two-step process to create your zFS aggregates via JCL. First, you define the aggregate and then you format it. In order to format the aggregate, you use the IOEAGFMT utility. Until now, in order to run IOEAGFMT, you needed either a UID of 0 or READ authority to the SUPERUSER.FILESYS.PFSCTL resource profile in the UNIXPRIV class. On the other hand, all you needed to create an HFS was ALTER authority to the data set profile.

Starting with z/OS V1R9, zFS will work just like HFS and allow users with ALTER authority to the data set profile to run the IOEAGFMT utility against that data set. This enhancement is also rolled back to z/OS releases V1R7 and V1R8.

Another zFS utility, IOEAGSLV, is also included in this change. Users with UPDATE authority to the data set profile now will be able to run this utility.

Note that, in actuality, all that is required to run the IOEAGFMT utility is UPDATE authority to the data set profile. However, since ALTER authority is required to define a VSAM linear data set and to set the zFS bit in the catalog, we will say that overall you need ALTER authority to create a zFS, just like you do to create an HFS.

There are other ways to create zFS aggregates as well. For instance, you can use the ISHELL panels or the **zfsadm** shell commands. Both of these methods would then use the zFS APIs to define and format the zFS aggregate. Note that, as of today, the zFS APIs are not changed as part of this enhancement. In order to format a zFS aggregate using these APIs, you still need a UID of 0 or READ authority to the SUPERUSER.FILESYS.PFSCTL profile in the UNIXPRIV class. Our team opened Marketing Requirement MR0608072541 to request that the zFS APIs support this change as well.

New aggregate full message from zFS

One of the zFS RAS enhancements in z/OS V1R9 is a new file system full message. We wanted to mention it here so that you are not surprised by it since it is displayed regardless of the aggregate full option you configured in your environment. The message is:

```
IOEZ00551I Aggregate AggrName ran out of space.
```

It will be issued no more than every 10 minutes for the same aggregate. If dynamic aggregate grow is on, zFS will attempt to grow it.

Chapter 9. Using the IBM WebSphere Business Integration family of products

The IBM WebSphere MQ (formerly MQSeries®) family of products forms part of the newly re-branded WebSphere Business Integration portfolio of products. These products are designed to help an enterprise accelerate the transformation into an on demand business.

The following topics describe our experiences:

Using WebSphere MQ shared queues and coupling facility structures

Using Websphere MQ, programs can talk to each other across a network of unlike components, including processors, operating systems, subsystems, and communication protocols, using a simple and consistent application programming interface.

We migrated our WebSphere for z/OS queue managers from V5.3.1 to V6.0. Much of our discussion here focuses on our experience with the usage and behavior of the coupling facility structures that support shared queues as well as using shared channels in a distributed environment with queue managers running V5.3.1. As we continue our testing with WebSphere MQ V6.0 we will start to describe our experiences here as well as in future releases of the test report.

We used information from the following sources to set up and test our shared queues:

- *WebSphere MQ for z/OS System Administration Guide*, SC34-6053 and *WebSphere MQ for z/OS System Setup Guide, SC34-6583*, for information about recovery from DB2, RRS, and CF failures. This document is available from the WebSphere Business Integration library at www.ibm.com/software/integration/websphere/library/.
- *WebSphere MQ in a z/OS Parallel Sysplex Environment*, SG24-6864, available from IBM Redbooks at www.ibm.com/redbooks/
- *WebSphere MQ Queue Sharing Group in a Parallel Sysplex Environment*, REDP-3636, available from IBM Redbooks at www.ibm.com/redbooks/

Our queue sharing group configuration

We currently have two queue sharing groups: one with three members and another with four members. The smaller queue sharing group is for testing new applications or configurations before migrating them to our production systems. The queue sharing groups each connect to different DB2 data sharing groups. This discussion will focus on the four-member production queue sharing group. All of the queue managers in the group run WebSphere MQ for z/OS Version 6.0.

Managing your z/OS queue managers using WebSphere MQ V6 Explorer

Websphere MQ V6 now offers an extensible Eclipse-based graphical configuration tool which replaces the Windows-based MQ Explorer. The Websphere MQ V6 Explorer is supported on both Windows® and Linux operating systems.

This tool, in conjunction with SupportPac MO71, has provided us with the ability to monitor as well as perform remote administration and configuration of our entire MQ network. The queue manager being managed does not have to be running WebSphere MQ V6 except when the queue manager is running on z/OS. If you wish to manage your z/OS queue managers using WebSphere MQ V6 Explorer and security is enabled on these queue managers you be will required to install refresh pack 6.0.1.1 or higher. This is because userids on z/OS are validated by RACF security and should be in uppercase. Without refresh pack 6.0.1.1, WebSphere MQ V6 Explorer transmits the userid to the queue manager in lowercase and subsequently the connections are rejected by RACF.

Our coupling facility structure configuration

We defined our MQ coupling facility structures to use three coupling facilities(CF1, CF2 and CF3) as defined in the preflist in the structure definitions. (See "Coupling facility details" on page 8 for details about our coupling facilities.)

The following is the structure definition for our CSQ_ADMIN structure:

```
STRUCTURE NAME(MQGPCSQ_ADMIN)
          INITSIZE(18668)
          MINSIZE(18668)
          DUPLEX(ENABLED)
          SIZE(20480)
          ALLOWAUTOALT(YES)
          PREFLIST(CF3,CF2,CF1)
          REBUILDPERCENT(1)
          FULLTHRESHOLD(85)
```

We also have the following five message structures defined to support different workloads:

- MSGQ1 — for the batch stress workload
- CICS — for the CICS bridge application
- EDSW — for the IMS bridge application
- WMQI — for the WebSphere Message Broker applications
- BOOK – for our BookStore workload (uses DB2, WMQ and Websphere Application Server)

The following is the structure definition for the message structure that supports the MQ-CICS bridge workload:

```
STRUCTURE NAME(MQGPCICS)
          INITSIZE(10240)
          DUPLEX(ENABLED)
          SIZE(20480)
          ALLOWAUTOALT(YES)
          PREFLIST(CF2,CF3,CF1)
          REBUILDPERCENT(1)
          FULLTHRESHOLD(85)
```

The other four message structures are defined similarly, except for the sizes. All of the structures are enabled for duplexing.

We chose to create multiple message structures in order to separate them by application. That way, if there is a problem with a structure, it will not impact the other applications. However, this is not necessarily the recommended approach from a performance perspective. See the Redbook Paper *WebSphere MQ Queue Sharing Group in a Parallel Sysplex Environment* for more information.

The CICS, EDSW, WMQI, BOOK and MSGQ1 structures are recoverable and backed up daily.

Recovery behavior with queue managers using coupling facility structures

We conducted the following types of test scenarios during our z/OS release testing:

- CF structure errors
- CF structure duplexing and moving structures between coupling facilities
- CF-to-CF link failures
- MQ CF structure recovery

During these tests, we monitored the behavior of the MQ queue managers as well as the behavior of applications that use shared queues.

Queue manager behavior during testing

We observed the following behavior during our test scenarios:

CF structure errors: With the MQ CICS bridge workload running, we used a local tool to inject errors into the coupling facility structures. When we injected an error into the MQ administrative structure, the structure moved to the alternate coupling facility, based on the preflist, as expected. Throughout the test, the CICS bridge workload continued to run without any errors.

CF structure rebuild on the alternate coupling facility: With system-managed CF structure duplexing active and a shared queue workload running, we issued the SETXCF STOP,REBUILD command to cause XCF to move the MQ structures to the alternate coupling facility. The queue manager produced no errors and the application continued without any interruption.

We also tested recovering into an empty structure. We first issued the SETXCF FORCE command to clear the structure, followed by the RECOVER CFSTRUCT(CICS) TYPE(PURGE) command. Again, the structure recovered with no errors.

Additional experiences and observations

MQ abends during coupling facility failures: Although coupling facility failures are extremely rare under normal operations, we induce many failures in our environment in the course of our testing. When coupling facility failures occur which have an impact on WebSphere MQ, such problems generally manifest themselves as MQ dumps with abend reason codes that start with 00C51*nnn*. Many of these are actually coupling facility problems or conditions that result in MQ having a problem and are not necessarily MQ problems in their own right. When such abends occur, we suggest that you analyze the system log for any IXC or IXL messages that might indicate a problem with a coupling facility.

Intra-group queuing: We have all members of the queue sharing group set up for intra-group queuing. This was done by altering the queue manager to enable

intra-group queuing. SDSF makes use of the SYSTEM.QSG.TRANSMIT shared queue for transmitting data between SDSF servers instead of the cluster queues. It continues to use the cluster queues and channels for members not in the queue sharing group. Currently all systems in our sysplex have the SDSF MQ function enabled so job output for one system can be viewed from any other system in the sysplex.

Effects of DB2 and RRS failures on MQ: We also tested how MQ reacts when DB2 or RRS become unavailable. The following are some of our observations:

- APAR PQ77558 fixes a problem with MQ V5.3.1 when RRS is cancelled while the queue manager is running.
- When DB2 or RRS become unavailable, the queue manager issues an error message to report its loss of connectivity with DB2 and which subsystem is down. An example of such messages could be:

  ```
  CSQ5003A !MQJA0 CSQ5CONN Connection to DB2 using DBWG pending, no active DB2
  CSQ5026E !MQJA0 CSQ5CONN Unable to access DB2, RRS is not available
  ```

  When DB2 becomes available again, MQ issues a message to report that it is again connected to DB2. For example:

  ```
  CSQ5001I !MQJA0 CSQ5CONN Connected to DB2 DBW3
  ```
- MQ abend reason codes that indicate a DB2 failure start with 00F5*nnnn*.

Notes about MQ coupling facility structure sizes:

- All of our MQ coupling facility structures are defined to allow automatic alter (by specifying ALLOWAUTOALT(YES) in the structure definitions in the CFRM policy), whereby XCF can dynamically change the size of a structure, as necessary. This is beneficial because it allows XCF to automatically increase the size of a message structure as needed to hold more messages.
- When we first defined the CSQ_ADMIN structure, we made it 10000K bytes in size. Our original sizing was based on the guidelines in *WebSphere MQ for z/OS Concepts and Planning Guide*, GC34-6051. However, we have since migrated to a higher CFCC level and increased the number of queue managers in the queue sharing group, which increases the size requirement for the CSQ_ADMIN structure. As a result, the queue manager recently failed to start because the CSQ_ADMIN structure was too small and issued the following message:

  ```
  CSQE022E !MQJA0 Structure CSQ_ADMIN unusable, size is too small
  ```

  We used the SETXCF START,ALTER command to increase the size of the structure. The following is an example of the command we issued:

  ```
  SETXCF START,ALTER,STRNAME=MQGPCSQ_ADMIN,SIZE=16000
  ```

  Accordingly, we also increased the value of INITSIZE() and MINSIZE() for CSQ_ADMIN in the CFRM policy up to 18668 to accommodate the increase in usage.

Running WebSphere MQ implemented shared channels in a distributed-queuing management environment

We implemented shared channels within the larger of our two queue sharing groups to bolster our distributed-queuing management (DQM) environment. Previously, we have had a DQM workload that exercised distributed messaging using MQ channels that provided an environment to test channel functionality such as SSL, as well as more general testing such as load stress. We modified the underlying DQM environment to utilize both shared inbound and shared

outbound channels without having to change the workload application. We are now able to handle higher amounts of inbound messages from remote MQ clients and, at the same time. provide transparent failover redundancy for those inbound messages.

Our MQ ″clients″ are in fact full MQ servers on distributed platforms such as Linux and Windows 2000.

Our shared channel configuration

The following sections describe the configuration of our shared inbound and outbound channels. We used information in *WebSphere MQ Intercommunication*, SC34-6059, to plan our configuration.

Shared inbound channels

We decided to implement the shared channel environment on our sysplex using TCP/IP services because our distributed DQM clients are mainly TCP/IP clients. All queue managers in the queue sharing group were configured to start group listeners on the same TCP port (1415), as described in the MQ intercommunication guide.

Example: The following is an example of the command to start group listeners on TCP port 1415:

```
START LISTENER INDISP(GROUP) PORT(1415)
```

The MQ intercommunication guide describes how the group listener port maps to a generic interface that allows the queue sharing group to be seen as a single network entity. For our DQM environment, we configure the Sysplex Distributor service of z/OS Communications Server to serve as the TCP/IP generic interface. This is a slight departure from the intercommunication guide, which utilizes DNS/WLM to provide the TCP/IP generic interface. VTAM generic resources is another available service that can provide the generic interface for channels defined using LU6.2 connections.

Example: The following is an example of our Sysplex Distributor definition for TCP port 1415:

```
VIPADYNAMIC
VIPADEFINE MOVEABLE IMMED 255.255.255.0  192.168.32.30
VIPADISTRIBUTE DEFINE 192.168.32.30 PORT 1415
DESTIP 192.168.49.30 192.168.49.32 192.168.49.33 192.168.49.38
ENDVIPADYNAMIC
```

We added this definition to the TCP/IP profile of one of our queue sharing groups (in this case 192.168.49.32), but it can be added to any TCP/IP host within the sysplex in which the queue sharing group resides. The IP addresses listed for DESTIP are the XCF addresses of the queue managers in our queue sharing group. The remote client can then specify 192.168.32.30 (or, correspondingly, the host name MQGP.PDL.POK.IBM.COM, which maps to the IP address in our DNS server for our 192.168.xx.xx LAN) on its sender channel, which then causes the receiver channel start to be load-balanced using the WLM mechanisms of Sysplex Distributor.

Example: The following is an example of our definitions for the remote sender channel and the local receiver channel:

```
DEFINE CHANNEL(DQMSSL.CSQ9.TO.MQGP) +
REPLACE +
CHLTYPE(SDR) +
```

```
XMITQ(DQMMQGP.QSG.XMITQ) +
TRPTYPE(TCP) +
DISCINT(10) +
CONNAME('MQGP.PDL.POK.IBM.COM(1415)') +
SSLCIPH(TRIPLE_DES_SHA_US) +
DESCR('DQM SDR CHANNEL TO SHARED RCVR CHANNEL ON MQGP')

DEFINE CHANNEL(DQMSSL.CSQ9.TO.MQGP) +
REPLACE +
CHLTYPE(RCVR) +
QSGDISP(GROUP) +
TRPTYPE(TCP) +
SSLCAUTH(REQUIRED) +
SSLCIPH(TRIPLE_DES_SHA_US) +
DESCR('SHARED RCVR CHANNEL FROM J90 FOR DQM')
```

Note that QSGDISP(GROUP) specifies that a copy of this channel is defined on each queue manager in the queue sharing group. This allows the inbound channel start request to be serviced by any queue manager in the queue sharing group. At this point, messages can be placed on application queues that are either shared or local to the queue manager (as long as they are defined on each queue manager in the queue sharing group, specifying QSGDISP(GROUP) in the definitions).

Shared outbound channels

The MQ intercommunication guide states that an outbound channel is a shared channel if it moves messages from a shared transmission queue. Thus, we defined a shared transmission queue for our outbound channels, along with an outbound sender channel with a QSGDISP of GROUP. This enables the queue managers in the queue sharing group to perform load-balanced start requests for this channel.

Example: The following is our definition for the shared transmission queue:

```
DEFINE QLOCAL(DQMCSQ9.QSG.XMITQ) +
REPLACE +
STGCLASS(DQMSTG) +
DESCR('SHARED XMITQ QUEUE FOR DQM TO J90') +
QSGDISP(SHARED) +
MAXDEPTH(2000) +
TRIGGER +
TRIGDATA(DQMSSL.MQGP.TO.CSQ9) +
INITQ(SYSTEM.CHANNEL.INITQ) +
USAGE(XMITQ) CFSTRUCT(MSGQ1)
```

Example: The following are our definitions for the local sender channel and the remote receiver channel:

```
DEFINE CHANNEL(DQMSSL.MQGP.TO.CSQ9) +
REPLACE +
CHLTYPE(SDR) +
XMITQ(DQMCSQ9.QSG.XMITQ) +
QSGDISP(GROUP) +
TRPTYPE(TCP) +
DISCINT(15) +
CONNAME(J90EIP.PDL.POK.IBM.COM) +
SSLCIPH(TRIPLE_DES_SHA_US) +
DESCR('SHARED SDR CHANNEL TO J90 FOR DQM')

DEFINE CHANNEL(DQMSSL.MQGP.TO.CSQ9) +
REPLACE +
CHLTYPE(RCVR) +
TRPTYPE(TCP) +
SSLCAUTH(REQUIRED) +
SSLCIPH(TRIPLE_DES_SHA_US) +
DESCR('DQM RCVR CHANNEL FROM SHARED SDR CHANNEL ON MQGP')
```

Enabling WebSphere MQ Security

We recently went through the task of enabling MQ security for the z/OS queue managers in our zPET environment. WebSphere MQ provides an interface to an external security manager which, in our case, is Resource Access Control Facility (RACF). When we decided to enable security for our queue managers, we took a step back to determine the best approach for our environment. Our simple approach to controlling security was to use queue-sharing group level of security for our queue managers that were members of a queue-sharing group and queue manager level of security for the rest of the queue managers in our environment which are not members of queue-sharing groups.

Referencing the 'System Setup Guide' section "Using RACF classes and profiles" we first verified that the WebSphere MQ classes were activated in RACF. As in most customer environments we then used our 'test plex' as our starting point for enabling MQ security. Our 'test plex' consists of 3 z/OS images each running a queue manager at V6.0. These 3 queue managers are all members of the queue-sharing group MQGT. Since all 3 queue managers are members of the same queue-sharing group we decided to use queue-sharing group level of security. We started by defining a basic set of profiles to each of the WebSphere MQ classes.

Reference material

We found the following reference material useful when working with WebSphere MQ Security:

- WebSphere MQ for z/OS Security (Technical Conference) which is a good overview located at:

  `http://www.gse.org.uk/wg/racf/docs/apr2005/GSE-%20WebSphere%20MQ%20zOS%20Security.pdf`
- **WebSphere Message Broker (WMB):** which outlines the necessary authority required by the broker. Search for "Authorization required" and then select "Summary of required access (z/OS)" at:

  `http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r0m0/index.jsp`
- **WebSphere MQ Explorer:** which outlines the necessary authority required by the MQ Explorer. Search for "Authorization to use WebSphere MQ Explorer" at:

  `http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp`
- **SDSF:** The following links document the necessary authority required by SDSF:
  - Communications:

    `http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/BOOKS/ISF4CS50/3.7?SHELF=ISF4BK50&DT=20050707140821`
  - **WebSphere:**

    `http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/BOOKS/ISF4CS50/7.29?SHELF=ISF4BK50&DT=20050707140821`
  - **SDSF Customization Wizard:** provides assistance in defining security for SDSF's use of MQ:

    `http://www-03.ibm.com/servers/eserver/zseries/zos/wizards/sdsf/sdsfv1r1/`

Once we had the basic profiles defined we started to enable security for each queue manager one at a time, resolving problems as they arose. We used RACF groups to grant authorities instead of individual userids which should make maintaining this security much easier. After enabling security for our 'test plex' we moved on to our 'production plex'. Our 'production plex' consists of 10 z/OS images each running a queue manager at V6.0. Of these 10 queue managers 4 of them are members of the same queue-sharing group MQGP. For the 4 queue managers that are members of a queue-sharing group we implemented security using the queue-sharing group level of authority. For the other queue managers we implemented the queue manager level of security.

Problems encountered

Following are some of the problems we encountered:

1. **WebSphere V6 Explorer:**
   a. After enabling security for our z/OS queue managers our connection to these queue managers using the WebSphere MQ Explorer were rejected with the following error:

   ```
   ICH408I USER(dodaro ) GROUP(        ) NAME(???                 )  932
      LOGON/JOB INITIATION - USER AT TERMINAL          NOT RACF-DEFINED
    IRR012I  VERIFICATION FAILED. USER PROFILE NOT FOUND.
   ```

   The userid was being sent to the host in 'lower case' and RACF was rejecting it. We installed fix pack 6.0.1.1 (U200247) for WebSphere MQ V6.0 to resolve this problem.

   b. With security enabled for our z/OS queue managers our connection was rejected with the following error:

   ```
   ICH408I USER(DODARO) GROUP(SYS1    ) NAME(####################) 015
     MQGT.AMQ.BF0F023EF3019DB9 CL(MQQUEUE )
     PROFILE NOT FOUND - REQUIRED FOR AUTHORITY CHECKING
     ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
   ```

   The queue being created was using the incorrect prefix 'AMQ.**' instead of 'AMQ.MQEXPLORER.**'. APAR IC50201 will resolve this issue.

2. **Mixed case' queue names:** After enabling security in our zPET environment we ran into a situation trying to access one of our queues. WebSphere MQ supports 'mixed case' for their queue names. We had a queue named 'Trade3BrokerTestQueue'. When we attempted to access this queue we received the following racf error:

   ```
   ICH408I USER(WAS5SSR3) GROUP(WASSRGP ) NAME(WAS 5 APPSVR SR 3
     MQGT.Trade3BrokerTestQueue CL(MQQUEUE )
     PROFILE NOT FOUND - REQUIRED FOR AUTHORITY CHECKING
     ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
   ```

   RACF currently does not allow defining 'mixed case' profiles for the MQ classes. To get around this situation we created a profile named 'MQGT.T*' and granted the necessary authority to this profile. Until RACF supports 'mixed case' profiles we would suggest that if you use 'mixed case' that your queue name is prefixed with enough characters in 'upper case' (for example TRADE3BrokerTestQueue} which will allow you to properly protect your queues.

3. **WebSphere Message Broker and WebSphere Application Server:** After enabling security for our z/OS queue managers we experienced problems when connecting to our queue managers from these applications when the userid being sent to the host was in 'lower or mixed case' and subsequently was rejected by RACF. This was the case with the WMB toolkit running on Windows and connecting to z/OS config mgr.. Here we changed the userid on Windows to be in 'uppercase'. This was also the case for WebSphere Application Server when the JMS resource was defined using a 'lowercase' userid causing the listener not to start. Again, here we were able to get around this problem by changing the JMS resource definition in WebSphere Application Server to use an 'uppercase' userid.

   ```
   MQJMS2013: invalid security authentication supplied for MQQueueManager   at startup.

   CSQ8MSTR has: ICH408I USER(setup   ) GROUP(        ) NAME(???             )
   LOGON/JOB INITIATION - USER AT TERMINAL          NOT RACF-DEFINED
   ```

Migrating to WebSphere Message Broker Version 6

Before migrating to Websphere Message Broker (WMB) V6 our WBIMB configuration consisted of three brokers at the WebSphere Business Integration Message Broker V5 level on one sysplex and two brokers at the same level on another sysplex. We used the WBIMB V5 toolkit on Windows connecting to a Configuration Manager which was also on Windows. We migrated all of our brokers to WMB V6, created a new broker, and added a z/OS Configuration Manager to one sysplex while the other one is still using the old Configuration Manager in our Windows machine. The option to have a z/OS configuration manager is new with WMB V6.

Changes from WBIMB V5 to WMB V6

The following are some of the changes we had to make from WBIMB V5 to WMB V6.

Directory structure changes

WMB V6 added a new HOME directory separate from the COMP directory used in WBIMB V5. Our new directory structure looks as follows:

```
/wmb60/basecode    contains the product code
/wmb60/COMP        contains a directory for each broker or configmgr
/wmb60/HOME        contains a directory for each broker and config mgr which has files bipprof and ENVFILE
```

Each of the above directories is mounted off of a separate ZFS filesystem.

DB2 DSNAOINI file changes

The WMB V6 started task JCL uses the *dsnaoini* from a dataset instead of using the one in the broker directory as WBIMB V5 did.

/wmb60/HOME/CSQ2BRK/bipprof has a statement:

```
export DSNAOINI=//\'WMB.CSQ2BRK.DSNAOINI\(BIPDSNAO\)\'
```

The broker ENVFILE contains the statement:

```
DSNAOINI=//'WMB.CSQ2BRK.DSNAOINI(BIPDSNAO)'
```

This points to the z/OS dataset member to get the values it needs.

We followed the migration instructions in the WMB V6 Information Center

```
http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r0m0/index.jsp
```

See the section titled ″Migrating from Version 5.0 products″ for the WebSphere Message broker product. We migrated a broker first, then the toolkit, and did the configuration manager last.

XML changes

The new WMB V6 broker requires the XML Toolkit for z/OS, Pgm 5655-J51. This was installed and referenced in the broker bipprof file XMLTOOLKIT=/ixm/ixm/IBM/xml4c-5_5 as well as in the ENVFILE.

Broker migration

Some of the things to watch out for with the Broker migration are:

- Be sure to never edit the files in the broker registry directory. If changes need to be made use **printf ″changed value″ > filename** . Editing can often add CR or LF characters which the broker does not handle well.

- When migrating, the component directory is the previous version's component directory. The HOME directory is new for WMB V6.

As a pre-migration task we backed up our broker databases and toolkit workspace data. We also backed up the component directories for the brokers. Then we followed the steps outlined in the section ″Migrating from WebSphere Business Integration Message Broker Version 5.0 to WebSphere Message Broker Version 6.0″ sub-topic ″Migrating a Version 5.0 broker to Version 6.0 on z/OS″ found at:

`http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r0m0/index.jsp`

Note: The Unix System Services environment variables of the userid running the migration jobs will be copied to the broker ENVFILE in the HOME directory. Be careful and review the ENVFILE to be sure you don't have variables set that you don't want for the broker.

We edited the /wmb60/HOME/CSQABRK/*ENVFILE* to remove all entries it added from */u/lorain0/.profile*. The jobs were run from userid *lorain0*.

All migration jobs ran successfully.

Toolkit migration

On Windows we backed up the WBIMB databases *WBICMDB* and *DWCTRLDB*. Then we used the '**export**' function in the wbimb v5 toolkit to save all projects and create a file structure for them.

We then ran the setup.exe for the new WMB V6 Toolkit. The install was successful and the toolkit was able to connect to the WBIMB V5 Configuration Manager as well as the V5 (not yet migrated) and V6 brokers on z/OS.

Configuration Manager migration on Windows

The following scripts were run as documented in the WMB V6 Information Center:

- *mqsimigratecomponents -c configmgr* pre-check (Note: Don't use the config mgr name.)
- *mqsimigratecomponents configmgr* do the migration
- *mqsimigratecomponents configmgr* do the migration

  These all succeeded so we then started the config mgr
- *mqsistart configmgr*

The conf mgr started successfully but when we started the toolkit it failed to connect to the new configmgr. The event log had:

```
( ConfigMgr ) Unexpected exception in ConfigurationManager class 'initialize' method; exception text:
 ''java.lang.NoSuchFieldError: msgToken'', ''msgToken''.   An exception was caught by the ConfigurationManager class
 'initialize' method while the Configuration Manager was being started or stopped. The exception text is:
 'java.lang.NoSuchFieldError: msgToken', 'msgToken'.
```

We found an IBM technote at:

`http://www.ibm.com/support/docview.wss?rs=849&context=SSKM8N&dc=DB520&uid=swg21229211&loc=en_US&cs=UTF-8&lang=en`

describing this error which says:

Problem

Configuration Manager start up fails with BIP1002E. java.lang.NoSuchFieldError: msgToken exception on Configuration Manager start up.

Cause

This problem occurs if the Config Manager is connected to a WebSphere® MQ V6 queue manager, but does not have the MQ Java™ Client classes located on the CLASSPATH used by the profile.

Solution

Add the following JARs to the CLASSPATH (all from inside the WMQ installation's lib directory):

```
providerutil.jar
com.ibm.mqjms.jar
ldap.jar
jta.jar
jndi.jar
jms.jar
connector.jar
fscontext.jar
```

We added each of the above jar files to the windows CLASSPATH as shown below:

```
C:\Program Files\IBM\WebSphere MQ\Java\lib\providerutil.jar;C:\Program Files\IBM\WebSphere
MQ\Java\lib\com.ibm.mqjms.jar;C:\Program Files\IBM\WebSphere MQ\Java\lib\ldap.jar;C:\Program Files\IBM\WebSphere
MQ\Java\lib\jta.jar;C:\Program Files\IBM\WebSphere MQ\Java\lib\jndi.jar;C:\Program Files\IBM\
MQ\Java\lib\jms.jar;C:\Program Files\IBM\WebSphere MQ\Java\lib\connector.jar;C:\Program Files\IBM\WebSphere
MQ\Java\lib\fscontext.jar;C:\Program Files\IBM\WebSphere MQ\Java\lib\com.ibm.mq.jar;
```

Then we tried a simple deploy which failed with the message: `the deployment message was addressed to a broker with a UUID 21f01d8e-0a01-0000-0080-ea101ddff920, but this does not match the UUID of the running broker (09fede6a-0a01-0000-0080-d8b172fb79c9).`

This was fixed by altering the universally unique identifier (UUID) using the Configuration Manager Proxy API Exercisor. Start the Configuration Manager Proxy API Exerciser. This is a sample application that demonstrates the capabilities of the Configuration Manager Proxy (a comprehensive Java interface that allows you to control broker domains programmatically). To start this application, we performed the following steps:

- On Windows, click **Start > IBM WebSphere Message Brokers 6.0 > Java Programming APIs > Configuration Manager Proxy API Exerciser**.
- Connect to the configmgr.
- Right click the broker name then select set UUID.
- Enter the new UUID value.

This is a sample application that demonstrates the capabilities of the Configuration Manager Proxy (a comprehensive Java interface that allows you to control broker domains programmatically).

Creating a z/OS configuration manager

We followed the instructions in the WebSphere Message Broker V6 Information Center titled ″Creating a Configuration Manager on z/OS″.

This task went very well with no problems.

Then we switched the WMB V6 Toolkit on Windows to connect to this new z/OS Configmgr. The userid used by the Windows machine was called *'mqtest'* in lowercase. When we deployed to the configmgr it received this RACF error:

```
SYSTEM.BROKER.CONFIG.QUEUE could not be opened (MQ reason code 2035 while trying to open the queue)
ICH408I USER(mqtest  ) GROUP(        ) NAME(???
  LOGON/JOB INITIATION - USER AT TERMINAL          NOT RACF-DEFINED
```

We had to change the Windows userid to be uppercase as lowercase user IDs will not work on z/OS when connecting to WebSphere MQ resources.

Then we had to authorize the Toolkit user to access the configmgr per the instructions in the WMB Info Center: See the section ″Ensure that your toolkit machine and user ID has the appropriate authorization on the z/OS Configuration Manager.″

In SDSF, grant access to your user ID. For this to work on all machines, enter:

```
'/F <started task name>,CA U=<userID>,A=YES,P=YES,X=F'
```

or to grant access to your user ID for a specific machine, enter:

```
'/F <started task name>,CA U=<userID>,A=YES,M=<machine name>,P=YES,X=F'
```

Verify the above by entering:

```
'/F <configmgrname>,LA
```

We then used the command:

```
'/F MQZ2CMGR,CA U=mqbroker/MQSTEST,A=YES,P=YES,X=F
```

The response message was +BIP8071I MQZ2CMGR 2 Successful command *completion*. We then recycled the configmgr and the toolkit then connected to configmgr successfully.

You cannot switch back and forth between configuration managers when deploying to the same brokers because of the UUID's that are assigned to the brokers. If you do try to deploy using a different configuration manager than was controlling the broker beforehand you will get an error message like:

```
BIP2045E: Broker CSQ2BRK running on WebSphere queue manager CSQ2 did not process
a deployment message, because it was addressed to a broker with a different identifier.
```

This message usually means that an attempt has been made to assign the broker to a second (or a reinitialized) Configuration Manager.

Each broker is identified by a universally unique identifier (UUID) which is allocated when the Message Brokers Toolkit or Configuration Manager Proxy creates a definition for the broker. When deployment occurs, a UUID check is made to help prevent accidental deployment of changes to brokers not under the control of the Configuration Manager. In this case, the deployment message was addressed to a broker with a UUID 7c2e2517-0d01-0000-0080-ae77adc1960f, but this does not match the UUID of the running broker (17794370-0701-0000-0080-c2dfca2e3733).

To switch to the new configmgr we used the Configuration Manager Proxy API Exercisor to change the UUID as described above.

Enabling higher availability for WebSphere MQ

With an ever increasing dependence on the infrastructure to perform critical business processes, the availability of this infrastructure is becoming increasingly more important. In an effort to provide high availability for our WebSphere MQ deployment, we recently converted our queue managers over to using unique application-instance dynamic virtual IP addresses (DVIPA).

Specifically, each of our queue manager CHINIT JCL streams now use the MODDVIPA utility to create (upon initialization) and delete (upon shutdown) unique application-activated dynamic virtual IP addresses. By utilizing application specific IP addresses, connectivity to each queue manager is dynamically re-established regardless of where in the sysplex the queue manager is restarted, either manually or via automatic restart manager (ARM).

To avoid failures when trying to create the DVIPAs prior to the DYNAMICVIPA block being processed, message `EZD1214I Initial Dynamic VIPA processing has completed` has been added to the TCP/IP initialization. This gives you a reliable message that can be used to automate the activation of an application-activated DVIPA. See APAR PK14941 for further details.

Here is a sample of the first step in our CHINIT JCL to create the DVIPA(*xx.xx.xx.xx*):

```
DVIPA(xx.xx.xx.xx):
//TCPDVP EXEC PGM=MODDVIPA,REGION=0K,TIME=1440,
//            PARM='POSIX(ON) ALL31(ON)/-p TCPIP -c xx.xx.xx.xx'
```

Here is a sample of the last step in our CHINIT JCL to delete the DVIPA(*xx.xx.xx.xx*):

```
//TCPDVP EXEC PGM=MODDVIPA,REGION=0K,TIME=1440,
//           PARM='POSIX(ON) ALL31(ON)/-p TCPIP -d xx.xx.xx.xx'
```

For additional information, see:

- *WebSphere MQ in a z/OS Parallel Sysplex Environment*, SG24-6864
- *z/OS Communications Server: IP Configuration Guide*, SC31-8775

MQCICS — WebSphere MQ-CICS adapter/bridge workload

Our MQCICS workload is a Java application that places a request message containing the name of a CICS transaction and required parameters. These transactions can be received by CICS either through the WebSphere MQ-CICS Bridge or the WebSphere MQ-CICS Adapter, depending on which process gets triggered by the request queue. The request queue is monitored by one or more CICS regions. After the request has been processed, the CICS region puts a message on the specified reply queue. Our Java application runs either through z/OS UNIX or WebSphere Application Server on z/OS.

When we first started running this workload, we had WebSphere MQ V5 Release 3.1, where each bridge monitor task needed its own request queue. This limitation was removed with WebSphere MQ V6.

For variety in our test environment, we configured a shared queue solution for transactions to be processed by the WebSphere MQ CICS Adapter. Meanwhile, we test our WebSphere MQ CICS Bridge setup with an MQ cluster configuration.

WebSphere MQ-CICS bridge monitor using clustered queues

In our first workload environment, we have one or more systems running the request applications to a Web front end being hosted by WebSphere Application Server. The queue where the requests are going to is being monitored by one WebSphere MQ-CICS bridge monitor on either of four queue managers. The CICS region that picks up the request then sends a reply to the queue manager being monitored by the client application. Figure 33 on page 116 demonstrates the cluster

environment.

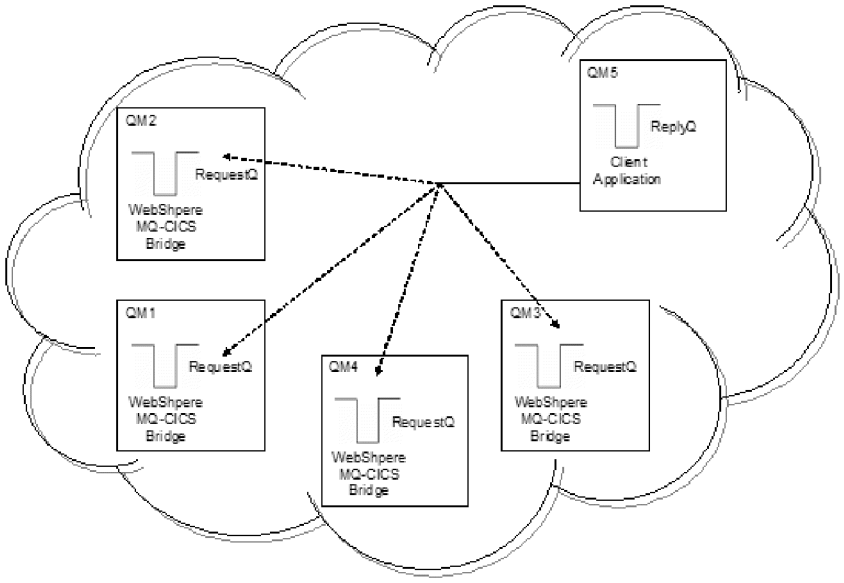


Figure 33. Our MQ cluster configuration for the WebSphere MQ-CICS bridge

WebSphere MQ-CICS adapter using shared queues

For our second workload environment shown in Figure 34 on page 117, we use a shared queue environment and the transactions are processed through the WebSphere MQ-CICS adapter. All three queues (request, reply, and initiation) are shared. All members of the queue sharing group have a CICS region monitoring the queue.

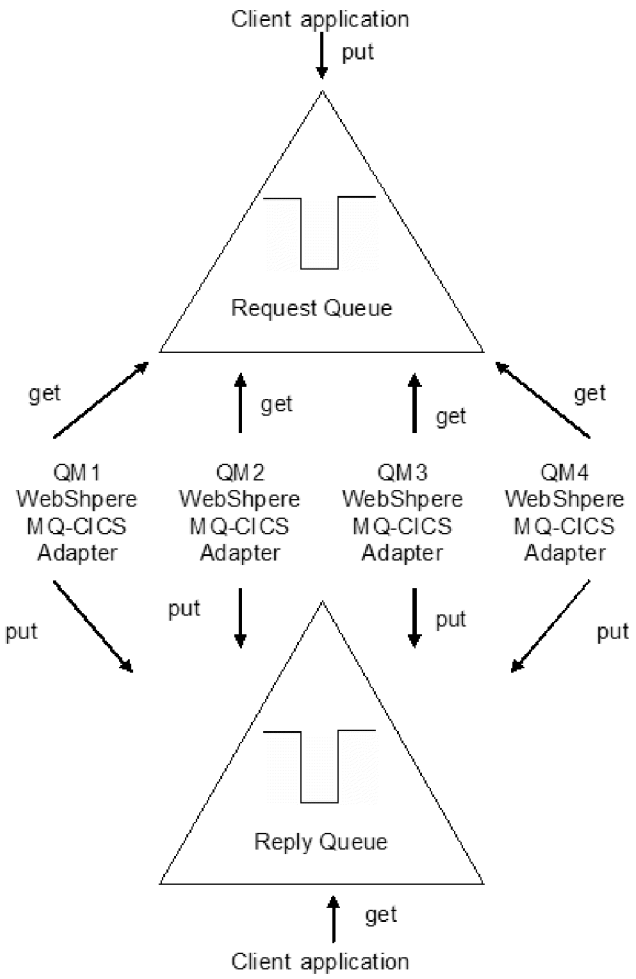


Figure 34. Our queue sharing group configuration for our WebSphere MQ-CICS adapter workload

Chapter 10. Using IBM WebSphere Application Server for z/OS

The following topics describe our experiences using IBM WebSphere Application Server for z/OS and related products. We have migrated most of our WebSphere Application Server for z/OS V6.0 cells to WebSphere Application Server for z/OS V6.1 on z/OS V1R9.

Note: References to WebSphere Application Server for z/OS V6.*x* appear in the text as "WebSphere for z/OS V6.*x*" or simply "V6.*x*."

About our z/OS V1R9 test environment running WebSphere Application Server

The following topics provide a level-set view of our current test environment and provide details about the changes we've made and our experiences along the way.

Our z/OS V1R9 WebSphere test environment

The following topics provide an overview of our z/OS V1R9 WebSphere test environment, including the set of software products and release levels that we run, the Web application configurations that we support, and the workloads that we use to drive them.

Our current software products and release levels

The following information describes the software products and release levels that we use on the z/OS platform and on the workstation platform.

Software products on the z/OS platform: In addition to the elements and features that are included in z/OS V1R9, our WebSphere test environment includes the following products:

- WebSphere Application Server for z/OS Version 6.1, service level cf50652.12
- WebSphere Application Server for z/OS Version 6.0.2, service level cf170648.05
- WebSphere Studio Workload Simulator V1.0
- WebSphere MQ for z/OS V6
- WebSphere Message Broker V6
- DB2 V9.1 with JDBC
- CICS TS 3.1
  - CICS Transaction Gateway (CICS TG) V6.1
- IMS V9 with IMS Connector for Java V9
  - IMS Connector for Java V9.1.0.1

Software products on the workstation platform: Software products on the workstation platform: On our workstations, we use the following tools to develop and test our Web applications:

- Rational® Application Developer Version 6.0.1.1
- IBM WebSphere Developer for zSeries Version 6.0.1
- WebSphere Studio Workload Simulator V1.0

Our current WebSphere Application Server for z/OS configurations and workloads

The following are our current WebSphere Application Server for z/OS configurations and workloads.

Configuration update highlights: We made the following updates to our test and production configurations:

- Migrated cells to WebSphere Application Server for z/OS V6.1
- Migrated to CICS Transaction Gateway (CICS TG) V6.1
- Added our zBank application (and J2EE server 7 for it)
- Implemented an enhancement of the zBank application, zCredit
- Security enhancements (TAM, TAI++, WebSeal on zLinux)
- Removed Node JH0 from P1 Cell (system removed from our test environment).

Our test and production configurations: In our environment, we have fully migrated most of our WebSphere for z/OS V6.0.2 cells to WebSphere for z/OS V6.1. Our current setup contains five cells: T1, T2, and T3 for our test systems, P1 for our WebSphere Application Server for z/OS production systems, and QP for WebSphere Application Server for z/OS applications used by MQ team. All cells are configured as network deployment cells. The QP cell continues to run using WebSphere for z/OS V6.0.2, while all others are now at V6.1.

Our T1 cell is configured as follows:

- Resides entirely on one of our test systems (Z1)
- Contains seven different J2EE servers, each running different applications (as described below)

Our T2 cell is configured as follows:

- Resides entirely on one of our test systems (Z2)
- Contains seven different J2EE servers, each running different applications (as described below)

Our T3 cell is configured as follows:

- Resides entirely on one of our test systems (Z3)
- Contains seven different J2EE servers, each running different applications (as described below)

Our P1 cell is configured as follows:

- Spans three production systems in our sysplex (J80, JB0 and JF0)
- Contains six different clusters, each of which spans all three systems. Each cluster contains four J2EE servers—one J2EE server per system.
- Each cluster corresponds to one of the single J2EE servers in our T1/T2 cell. Initially, we configure and deploy applications on a test J2EE server in the T1 and/or T2 cell and then deploy them to the corresponding server cluster in the P1 cell.

Our QP cell is configured as follows:

- Spans two production systems in our sysplex (JC0 and J90)
- Contains two different clusters, each of which spans both systems. Each cluster contains two J2EE servers—one J2EE server per system.

- Each cluster hosts various applications that connect WebSphere Application Server for z/OS to MQ as used by the MQ team.

Our Web application workloads: The following applications run in the J2EE servers on our T1, T2 and P1 cells:

- J2EE server 1 runs our workload monitoring application. The application accesses only z/OS UNIX System Services files.
- J2EE server 2 runs our bookstore application, accessing DB2 and WebSphere MQ
- J2EE server 3 runs the Trade6 application, accessing DB2 and WebSphere MQ
- J2EE server 4 runs our PETRTWDB2 application, accessing DB2
- J2EE server 5 runs our PETDSWIMS application, accessing IMS
- J2EE server 6 runs our PETNSTCICS application, accessing CICS

The following application runs in the J2EE Server on our T2 and T3 cells in addition to the above six applications:

- J2EE server 7 runs our zBank application used for security testing and accessing DB2

Figure 35 on page 122 shows the server address spaces in our P1 cell.

Note: The wsp1s1 cluster is not shown in the diagram.

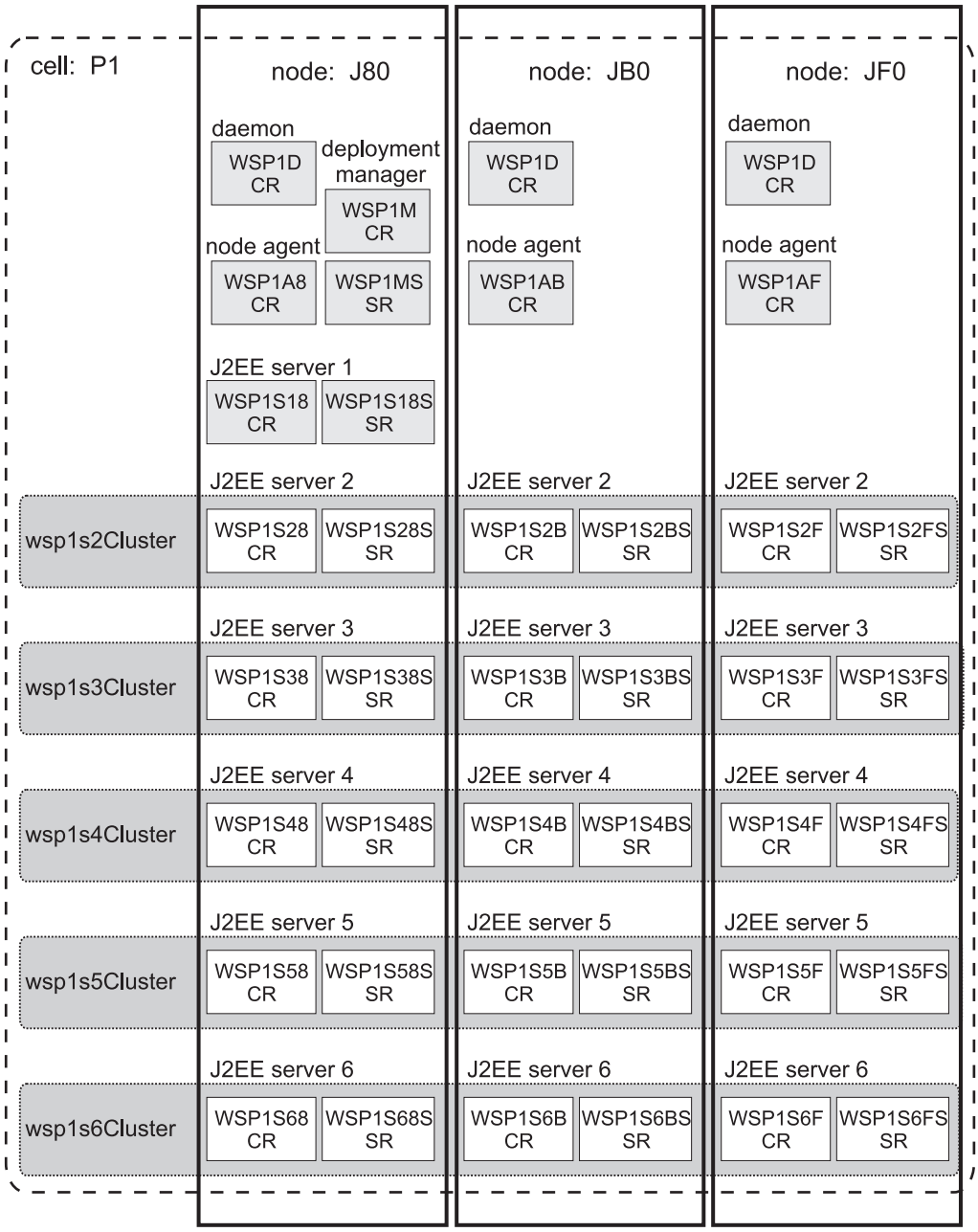


Figure 35. Our WebSphere for z/OS V6 configuration

About our naming conventions: After some experimentation, we settled upon a naming convention for our WebSphere setups. Our address space names are of the following format:

`WSccs[n]y[S]`

where:

WS The first two characters are always "WS" to identify a WebSphere resource.

cc Cell identifier:
- **T1** Test cell 1
- **T2** Test cell 2
- **P1** Production cell 1
- **QP** MQ Team Production cell

s[*n*] Server type. For J2EE server control regions and server regions, *n* is the instance number of the server within the node:
A Node agent
D Daemon
M Deployment manager
S*n* J2EE server control region, instance *n*

y System identifier:
1 Z1 (test)
2 Z2 (test)
8 J80 (production)
B JB0 (production)
F JF0 (production)

[S] Servant flag. This is appended to the name of a J2EE server control region to form the name of the associated servant region(s).

Example: The name WSP1S18S indicates a **W**eb**S**phere **p**roduction cell **1** J2EE server **s**erver region **1** on system J**8**0.

Server short names are specified in upper case. Server long names are the same as the short names, but are specified in lower case.

Other changes and updates to our WebSphere test environment

The following topics describe other changes and updates to our WebSphere test environment.

Migrating to WebSphere Application Server for z/OS V6.1

We have migrated most of our WebSphere Application Server V6.0.2 cells to V6.1. Overall, our migrations to V6.1 have been very smooth. The process is very similar to the migration from V5.1 to V6.0.2. It still requires a good bit of planning and work to migrate to V6.1 from V6.0.2. Careful review of all the latest documents in the WebSphere InfoCenter is highly recommended.

While we did have some problems in our initial V6.1 migrations, we successfully migrated from our WebSphere Application Server for z/OS V6.0.2 with service level CF180704 to WebSphere Application Server for z/OS V6.1 at service level CF50625. Many fixes have been included and it is recommended to apply the latest service updates, including those for the V6.0.2 configuration from which you are migrating, prior to starting.

One issue we ran into is now addressed in APAR PK48599. It was due to our setups using z/OS UNIX symbolic links within our WebSphere configurations and the configurations using IMS or CICS resource adapters. If your setups have neither, you need not worry about this. We used the local fix described in Tech Doc #21257063 (available at www.ibm.com/support/docview.wss?rs=404 &uid=swg21257063) during our migrations, but the formal fix for this APAR should be available by press time and is the recommended way to go.

The Legacy RRS connector for DB2 JDBC access is no longer supported on WAS V6.1. Many of the JDBC resources were still defined using this JDBC provider. To help migrate these to DB2 JDBC JCC resources, we used a utility available from the WebSphere Application Server support web pages. The utility is well documented, easy to use, and ran very well for us. See the "JDBC Migration White Paper and Utility for DB2 on z/OS" available at www.ibm.com/support/docview.wss?rs=404&context=SS7K4U&q1=RRS&uid=swg27007826&loc=en_US&cs=utf-8&lang=en. The utility can be run either against the Version 6.0.2 setup prior to migration to Version 6.1 or after on the Version 6.1 setup. We chose to update our provider prior to migrating to V6.1 to prevent errors after the migration.

See the following references for more information:

- WebSphere Application Server for z/OS V6.1 InfoCenter at publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp
- WebSphere Integration Test team's report on migrations to WebSphere Application Server V6.1 from various levels, which you can find at publibz.boulder.ibm.com/epubs/pdf/e0z1r100.pdf
- Also see the WebSphere Integration Test team's current report, which you can find at publibz.boulder.ibm.com/epubs/pdf/e0z1r111.pdf

Migrating to CICS Transaction Gateway V6.1

We made changes to the WebSphere environment when migrating to CICS Transaction Gateway (CICS TG) V6.1. See "Migrating to CICS Transaction Gateway V6.1" on page 73 for specific information about these changes.

Passing DB2 client information to the server

We tested the DB2-only methods provided by the DB2 Universal JDBC Driver that can be used to provide extra information about the client to the server. This can really help make your DB2 administrator's life a bit easier!

Passing client information to DB2 from WebSphere Application Server datasources

One of the common complaints we often hear from our DB2 database administrators is that they can't tell where a thread is coming from and what application it's from.

While Type 2 connections provide a bit more detail, such as the local address space initiating the connection, it still leaves our DBAs scratching their heads saying, "What app is that?" With Type 4 (TCP/IP) connections, it becomes even harder.

The DB2 Universal JDBC Driver provides DB2-only methods that you can use to provide extra information about the client to the server. WebSphere Application Server makes it easy to add these to your DB2 datasources.

For full details, see the DB2 Information Center at publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp. Look for the topic titled "Providing extended client information to the DB2 server with the DB2 Universal JDBC Driver" under "Advanced JDBC application programming concepts."

Table 10 on page 125 shows, from the above reference, the methods that you can use to pass additional client information.

Table 10. DB2 Universal JDBC driver methods for passing client information to the server

Method	Information provided
setDB2ClientUser	User name for a connection
setDB2ClientWorkstation	Client workstation name for a connection
setDB2ClientApplicationInformation	Name of the application that is working with a connection
setDB2ClientAccountingInformation	Accounting information

For our J2EE applications running in WebSphere Application Server (managed servers), these methods can be set as properties on the DB2 datasource.

Prior to setting these priorities, a DB2 DISPLAY THREAD command showed the following for a connection from this server:

```
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
 SERVER   RA *    22 db2jcc_appli DB2USR    DISTSERV 011B 99099
  V437-WORKSTATION=Z2EIP.PDL.POK.IB, USERID=db2usr,
       APPLICATION NAME=db2jcc_application
  V445-G90C14A2.GB60.BFD666FD252E=99099 ACCESSING DATA FOR
   ::FFFF:9.12.20.162
```

Using the WebSphere Application Server admin console Web application, we set the custom priorities for the JDBC datasource for our application as shown in Table 11.

Table 11. Custom priorities that we set for the JDBC datasource for our application

Datasource property	Our value set	Our usage
clientWorkstation	`WST2S22_3`	Address space or workstation name initiating the connection. A suffix (_3) is used if the app/server has more than one datasource.
clientApplicationInformation	`zipSeriesStore`	Application using the datasource
clientAccountingInformation	`BookStoreEJBDB2Entity`	Datasource resource name (same as used in WebSphere Application Server admin console)

After the server was updated with the changes, the client information is now sent and the DISPLAY THREAD command now shows the following information:

```
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
SERVER   RA *    31 db2jcc_appli DB2USR    DISTSERV 011B 157752
 V437-WORKSTATION=WST2S22_3, USERID=db2usr,
      APPLICATION NAME=zipSeriesStore
 V445-G90C14A2.G6F9.BFD66C17E48A=157752 ACCESSING DATA FOR
  ::FFFF:9.12.20.162
```

The clientAccountingInformation is not shown when using the DISPLAY THREAD command unless you add the DETAIL option (for instance: DIS,THD(*),DETAIL). See “Example: Thread detail output for a Type 4 connection from WebSphere Application Server” on page 126 for an example showing the use of the DETAIL option.

Additional notes and experiences with passing DB2 client information

We have the following additional notes and experiences to share about passing DB2 client information:

- While we could set the clientUser property, we found this a bit confusing on the dis,thd side. In the following example, we set the clientUser property to BookStore_Search. The actual user ID that is used for the connection (SETUP) is displayed in the first line of the output. The value specified for the clientUser property shows up in the second line (USERID=BookStore_Search). This made it a bit more confusing as the clientUser property is not associated with any user ID and is only informational, but the display output has the USERID= shown along with this. Not setting this property, the actual user ID is displayed, so the two lines of output match.

```
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
SERVER   RA *    41 db2jcc_appli DB2USR    DISTSERV 011B 12627
 V437-WORKSTATION=WST2S22_3, USERID=BookStore_Search,
      APPLICATION NAME=zipSeriesStore
 V445-G90C14A2.G574.BFD671E125DA=12627 ACCESSING DATA FOR
  ::FFFF:9.12.20.162
```

- In many of our applications we have multiple datasources defined and used. We found it helpful to add some additional information to the clientWorkstation or clientApplicationInformation property to help discern between them, since these values show up in the DISPLAY THREAD output. In our examples, we have added a suffix to the clientWorkstation property that identifies to us the third resource defined for this application (WST2S22_3) . While this bit of information doesn't always help our DB2 administrators, it does help our WebSphere Application Server administrators.
- For standalone (non-managed) Java applications that access DB2, you need to code these methods to enable. See the sample code in the DB2 Information Center.

For further information, see the WebSphere Application Server Information Center at publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp. Look for the topic titled, "Passing client information to a database."

Example: Thread detail output for a Type 4 connection from WebSphere Application Server

The following is an example of the response from the DIS,THD(*),DETAIL command for a Type 4 connection from WebSphere Application Server, before setting any of the client*xxx* properties:

```
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
SERVER   RA *    48 db2jcc_appli DB2USR    DISTSERV 011B 99099
 V437-WORKSTATION=Z2EIP.PDL.POK.IB, USERID=db2usr,
      APPLICATION NAME=db2jcc_application
 V441-ACCOUNTING=JCC03010Z2EIP.PDL.POK.IBM.
       ',X'00'
 V436-PGM=NULLID.SYSLN200, SEC=1, STMNT=0
 V445-G90C14A2.GB60.BFD666FD252E=99099 ACCESSING DATA FOR
  (  1)::FFFF:9.12.20.162
 V447--INDEX SESSID           A ST TIME
 V448--(  1) 446:2912         W S2 0634511043933
```

The following is an example of the response from the DIS,THD(*),DETAIL command after setting the clientWorkstation, clientApplicationInformation, and clientAccountingInformation properties for the datasource:

```
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
SERVER   RA *    68 db2jcc_appli DB2USR    DISTSERV 011B 157752
 V437-WORKSTATION=WST2S22_3, USERID=db2usr,
      APPLICATION NAME=zipSeriesStore
 V441-ACCOUNTING=BookStoreEJBDB2Entity
 V436-PGM=NULLID.SYSLN200, SEC=1, STMNT=0
 V445-G90C14A2.G6F9.BFD66C17E48A=157752 ACCESSING DATA FOR
```

```
  (  1)::FFFF:9.12.20.162
 V447--INDEX SESSID          A ST TIME
 V448--(  1) 446:1785        W S2 0634511195815
```

Example: Thread detail output for a Type 2 connection from WebSphere Application Server

The following is an example of the response from the DIS,THD(*) command for a Type 2 connection from WebSphere Application Server, before setting any of the client*xxx* properties:

```
NAME     ST A   REQ ID          AUTHID   PLAN     ASID TOKEN
RRSAF    TD       4 WST2S22S    DB2USR    ?RRSAF   01DC 56848
```

The following is an example of the response from the DIS,THD(*) command after setting the clientWorkstation, clientApplicationInformation, and clientAccountingInformation properties for the datasource:

```
NAME     ST A   REQ ID          AUTHID   PLAN     ASID TOKEN
 RRSAF    TD       4 WST2S22S    DB2USR    ?RRSAF   01B0 20436
  V437-WORKSTATION=WST2S22_3, USERID=*,
       APPLICATION NAME=zipSeriesStore
```

The DIS,THD(*),DETAIL command displays the value of the clientAccountingInformation property:

```
NAME     ST A   REQ ID          AUTHID   PLAN     ASID TOKEN
 RRSAF    TD       4 WST2S22S    DB2USR    ?RRSAF   01B0 20436
  V437-WORKSTATION=WST2S22_3, USERID=*,
       APPLICATION NAME=zipSeriesStore
  V441-ACCOUNTING=BookStoreEJBDB2Entity
```

Installed TPC-R V3.3

We made changes to the WebSphere environment when we installed TPC-R V3.3 in our zPET environment. For specific information about these changes, see "Using TPC-R V3.3 in our zPET environment" on page 12.

Where to find more information

During our testing, we used documentation from several sources, listed below. They contain all of the documents that we have cited throughout the course of this chapter.

- IBM WebSphere Application Server for z/OS documentation, available at `http://www.ibm.com/software/webservers/appserv/zos_os390/library/`
- IBM WebSphere Application Server, Version 6.0 Information Center, available at publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp
- IBM DB2 Information Center, available at publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp
- IBM Techdocs (flashes, white papers, and others), available at www.ibm.com/support/techdocs/
- *Java 2 Platform Enterprise Edition Specification*, available at http://java.sun.com/products/j2ee/
- IBM CICS Transaction Gateway documentation, available at http://www.ibm.com/software/ts/cics/library/
- IBM HTTP Server for OS/390 documentation, available at http://www.ibm.com/software/webservers/httpservers/library/
- IBM WebSphere Studio Workload Simulator documentation, available at www.ibm.com/software/awdtools/studioworkloadsimulator/library/

Part 2. Linux virtual servers

This part describes the Linux on System z and Linux virtual server aspects of our computing environment.

We address such topics as:

- Any significant updates to our environment since our last test report
- Our current test efforts and results
- Where we're headed from here and what we hope to report on in upcoming test reports

Chapter 11. About our Linux virtual server environment

With this release of our test report, we discuss the implementation of the first phase of our reference systems management architecture, focusing on availability management. Availability management covers the policies, processes and tools whose main purpose is to keep a collection of systems available to its intended users.

If you have read our previous test reports, you already know that our environment employs a fairly large number of discreet virtualized Linux instances spread across several hardware platforms, exercising a diverse array of middleware products and high availability solutions. Yet, the team managing and exploiting this environment is very modest in size. This test report provides some of the essential information distilled from our lab operations. With these essential management tools in place, the operation of the Linux Virtual Server Platform Evaluation Test (LVS PET) laboratory environment can be extremely complex yet highly efficient at the same time.

More than ever, customers are demanding superior systems management tools for the enterprise and SMB environments. It is directly because of customer feedback that we have chosen to place such a heavy emphasis on systems management.

For the initial phase of the systems management investigation discussed here, the LVS PET team, with its close proximity with the z/OS Platform Evaluation Test (zPET) team, has implemented a number of products running in the z/OS and Linux environments. Specifically, during this phase we have investigated the following critical components that form the foundation of enterprise systems management:

- monitoring
- logging
- notifying
- problem determination
- automation

Some of the products we have tested thus far offer overlapping capabilities. Due to time constraints, we did not test all functionality of all product offerings. Instead, we chose to implement a subset of functionality from each product. We do not in any way intend to imply that any particular product is superior to any other but, instead, have opted to show a variety of solutions based on our actual test results. This test report serves to share our experiences and insights during our investigation.

Chapter 12. Centralized system logging

At the very core of any managed environment is a solid logging infrastructure which entails centralizing and consolidating the log information. Due to the overwhelming number of routine, warning, and error messages generated by Linux systems, a single Linux server can be difficult to keep up with. The management problem is only exacerbated when multiplied by a large number of production Linux systems in a large environment.

We will discuss our experience with implementing a centralized logging server using **syslog-ng** functionality found in modern Linux distributions deployed in conjunction with **logrotate**.

Using syslog-ng

The **syslog-ng** software, an alternate to **syslogd**, is the new generation of UNIX and Linux logging server designed for large installations where security and flexibility are of concern. Where security concerns are less of an issue, such as behind a firewall, **syslog-ng** can receive logs from all Linux systems over a TCP connection. However, where security is a concern, **syslog-ng** has the ability to receive logs over a secure SSL connection as well. **syslog-ng** provides flexibility by allowing you to determine how incoming logs are filtered and split, as well as offering granular control over the format in which the logs are written.

In addition to log centralization and consolidation, **syslog-ng** promotes good system hygiene by reducing the volume and size of logs that must be stored on each individual Linux server. When used in conjunction with a log rotation mechanism, discussed in a separate topic, you will be equipped to handle even the most verbose logs while maintaining compliance with data retention directives.

System administrators will appreciate the flexibility of **syslog-ng** filtering and logging capabilities. A single log generation source can be used in one of two ways:

- For system error and recovery analysis, verbose logs can be archived.
- For real time analysis, you may instead opt for highly filtered logs of special interest.

In our deployment, the method in which the numerous source message logs were archived was important to us. We determined that the best solution was to store the archived logs in a file structure of */year/month/day/* on our central logging server. In addition, we needed a naming convention within the file structure that would retain the identity of the source system. We chose to suffix these centralized logs with `-$HOST` to identify the origin. Storing the logs in this manor provided us not only with a method that would allow recalling a number of logs from a particular date rapidly for analysis, but also provided us the ability to prune our archives by day, month, or year with a single deletion command. Note that your central logging server can also make it easy to backup older centralized logs to long-term storage medium from a single host.

We found, for our environment, we can store over six months of logs on a single 3390 model device. Of course, your needs may vary and the best approach is to

experimentally determine your needs during your prototype phase of deployment. If possible, when staging your centralized logging server, feed it with realistic data, preferably from actual production hosts.

Setting up the log server — LITSLOG1

We chose to install our **syslog-ng** log server on SuSE Linux Enterprise Server 10 (SLES 10) Service Pack 1. This distribution enables **syslog-ng** by default after system installation. Since it was our intention to archive logs from multiple sources to a file structure that differs from the default, we needed to make a couple of alterations to the default **syslog-ng** configuration.

SLES 10 provides a source template for the **syslog-ng** configuration file located in /etc/syslog-ng/syslog-ng.conf.in. To update the **syslog** configuration, edit the template and then execute the **suseconfig –module syslog** command which generates the configuration file, /etc/syslog-ng/syslog-ng.conf, which is later processed by the **syslog-ng** service.

The following are the two updates we made to the template file:

1. We wanted the archive logs written to be very verbose. This required removing the console filter on the log statement. The log statement we used is shown below. We retained the original statement as a comment so we could reference it later if needed:

   ```
   # log { source(src); filter(f_console); destination(console); };
   log { source(src);  destination(console); };
   ```

2. Our logging needs required updating the destination messages statement to reflect the file structure we intended on using to archive the system logs. In addition, we needed updates to preserve source host identity for each log written. Both of these were accomplished with updates to the file section of the destination message statement, as follows:

   ```
   destination messages {
   file("/var/log/logs/$YEAR/$MONTH/$DAY/messages-$HOST"
   create_dirs(yes)); };
   ```

Isolating and managing space used for archiving logs was of a concern to us. We mitigated our concern by opting to mount a dedicated 3390 model 3 Linux-formatted device at /var/log/logs/. This is the reason we did not write our logs to the standard /var/log/ location, as is reflected in the example above. If our single 3390 model 3 device had proved to be too small, we could simply have replaced it with a larger capacity volume.

Note that, when initially deploying **syslog-ng** after altering the configuration file as shown above, we received an error when **syslog-ng** attempted to write out the logs. We had missed a critical step in our configuration that is worthy of special attention. By default, **syslog-ng** will not create new sub-directories and requires the **create_dirs(yes)** option on the **destination messages** statement. After adding this option to the configuration file, **syslog-ng** performed as expected. If your directories already exist you should not need this statement but we expect many custom installations will benefit from this option, as we did.

Enabling Linux to send logs to the LITSLOG1 log server

Enabling our Linux systems running the older **syslogd** to send their logs to our **syslog-ng** log server, LITSLOG1, was a trivial task. It simply required the addition of a single statement in the /etc/syslog.conf configuration file. At the top of the configuration file, we simply added the statement:

```
*.* @litslog1
```

No other alterations to the Linux systems were required.

Client side log management with logrotate

Now that we have a centralized logging infrastructure in place, it makes sense to try to manage the voluminous records that would otherwise remain as copies on the local guests. This is not just a convenience but a necessity in the enterprise. In fact, managing your logs effectively is critical to system operation. In many instances applications fail when disk space is full. Mission critical enterprises clearly need logging but how much data should be kept locally and for how long?

Unfortunately there is no single correct answer to this question. Different servers may have different needs and your corporate guidelines on data retention may come into play. No matter your situation, what is clear is that each system needs some sort of log management policy. Some servers may generate more voluminous logs than others. Luckily, there is a tool that is well suited to local log management that is provided with modern enterprise Linux distributions, namely **logrotate**.

Logrotate allows you to specify how large a log file can grow (via compression or limiting), how often log files are cleared or backed up, and when to remove stale backups. Furthermore, log files can be processed in this way on a daily, weekly, or monthly basis, as needed, assuming the file has not reached a specified maximum capacity first.

Logrotate is typically executed by the **cron** service, which provides execution control at specified intervals.

You may already have **logrotate** in action and not be aware of it. If you have ever noticed some oddly named logs in /var/log that have version numbers tacked on the normal log filename, or perhaps a version of a log file name with a numbered .gz extension, **logrotate** may be operating already. The former is a simple rotation where some number of logs are kept for a given log file, where each numbered version is a specified maximum capacity or duration of time. The latter is the result of using the same approach but with the added compression options in place.

The control of these logs is rather straightforward. **Logrotate** can operate with a single configuration file (controlling multiple log files), or you may opt for a configuration file per log approach. On most distributions, the configuration is specified in the /etc/logrotate.conf configuration file. Many distributions also include pre-configured **logrotate** options for common services like the Apache Web server in the /etc/logrotate.d directory.

The following sample logrotate.conf will be used to introduce the most common options:

```
weekly
rotate 2
create
include /etc/logrotate.d

/var/log/messages {
missingok
create 0664 root utmp
rotate 1

}

/var/log/syslog {

missingok

create 0664 root utmp
daily
}
```

Figure 36. Sample logrotate.conf configuration file

The first lines in the file act globally for **logrotate**. The value `weekly` indicates that files should be kept on weekly intervals, with two iterations kept at a time. That is, half-a-month's worth of logs will be kept in two versioned files representing a week each.

The directive block beginning with the /var/log/messages path contains the local overrides and options that act on the specified path. Should you want to operate on a different path, copy the block and replace /var/log/messages with the log file of interest. Note that any overrides placed between the curly braces ({}) will take precedence over global directives. For instance, in Figure 36, the path /var/log/syslog will be rotated in two iterations as specified globally, but /var/log/messages will be rotated in a single iteration. Also, each of the versions syslog entries will contain a daily log rather than weekly.

Based on the requirements of your infrastructure, you may need to customize the rules. The entire set of **logrotate** options can be found in the documentation shipped with your distribution. See **man logrotate** for more options, including compression and other techniques you may find useful.

If you are unsure if you are using **logrotate** or if you are deploying it for the first time, you may want to check your /etc/crontab for entries including **logrotate**. Without a **cron** job in place to start **logrotate**, it will not be operational. Luckily, most distributions set it up automatically when you install the **logrotate** package.

Using the dbginfo.sh script

A script is provided on both the SuSE and RedHat distributions called **dbginfo.sh**. On SuSE, this package is part of the s390-tools package; on RedHat, it is shipped as a component of s390utils.

This script is a producer of vital information that needs to be logged from time to time. In fact, this script captures a great deal of system status information useful for debugging. We chose to execute this script weekly on each of our production systems to establish a status baseline. You may chose to do so less or more often in your deployment as you see fit. The output of this script, a .tgz file, is relayed to our centralized logging storage server.

The major purpose of this script and its output is to enable users to look back for anomalies that might have been occurring in the environment, eventually leading to problems. Extracting one of the tgz files will give you an example of its contents:

```
[root@lithub tmp]# tar -xzf DBGINFO-2007-10-15-16-46-01.tgz
[root@lithub DBGINFO-2007-10-15-16-46-01]# ls -al
total 200
drwxr-xr-x   7 root root  4096 Oct 15 16:46 .
drwxrwxrwt  11 root root  4096 Oct 15 16:51 ..
-rw-r--r--   1 root root 39458 Oct 15 16:46 dbginfo.log
drwxr-xr-x   2 root root  4096 Oct 15 16:46 etc
drwxr-xr-x   3 root root  4096 Oct 15 16:46 lib
drwxr-xr-x   4 root root  4096 Oct 15 16:46 proc
-rw-r--r--   1 root root 20699 Oct 15 16:46 runtime.out
drwxr-xr-x   6 root root  4096 Oct 15 16:46 sys
-rw-r--r--   1 root root 65344 Oct 15 16:46 sysfsfiles.out
drwxr-xr-x   3 root root  4096 Oct 15 16:46 var
```

Figure 37. Example of the contents of the output file from **dbginfo.sh**

In the following example, we start with the output shown in Figure 37 and switch to various directories and display the contents of a given file. This is the type of information that is provided by **dbginfo.sh**:

```
[root@lithub DBGINFO-2007-10-15-16-46-01]# cd proc
[root@lithub proc]# ls -al
total 132
drwxr-xr-x   4 root root  4096 Oct 15 16:46 .
drwxr-xr-x   7 root root  4096 Oct 15 16:46 ..
-r--r--r--   1 root root   210 Oct 15 16:46 cpuinfo
drwxr-xr-x   2 root root  4096 Oct 15 16:46 dasd
-r--r--r--   1 root root   227 Oct 15 16:46 devices
-r--r--r--   1 root root   602 Oct 15 16:46 meminfo
-r--r--r--   1 root root    18 Oct 15 16:46 misc
-r--r--r--   1 root root   739 Oct 15 16:46 modules
-r--r--r--   1 root root   399 Oct 15 16:46 mounts
-r--r--r--   1 root root   199 Oct 15 16:46 partitions
-r--r--r--   1 root root   414 Oct 15 16:46 qeth
drwxr-xr-x  20 root root  4096 Oct 15 16:46 s390dbf
-rw-r--r--   1 root root 13583 Oct 15 16:46 slabinfo
-r--r--r--   1 root root  1737 Oct 15 16:46 sysinfo
-r--r--r--   1 root root   149 Oct 15 16:46 version

[root@lithub proc]# cat meminfo
MemTotal:      1023312 kB
MemFree:        320792 kB
Buffers:         93580 kB
Cached:         483824 kB
SwapCached:          0 kB
Active:         296936 kB
Inactive:       309020 kB
HighTotal:           0 kB
HighFree:            0 kB
LowTotal:      1023312 kB
LowFree:        320792 kB
SwapTotal:      511988 kB
SwapFree:       511988 kB
Dirty:             412 kB
Writeback:           0 kB
Mapped:          42892 kB
Slab:            74968 kB
CommitLimit:   1023644 kB
Committed_AS:    63808 kB
PageTables:        836 kB
VmallocTotal: 4293910528 kB
```

```
VmallocUsed:      6168 kB
VmallocChunk: 4293902184 kB

[root@lithub proc]# cat sysinfo
Manufacturer:         IBM
Type:                 2084
Model:                325
Sequence Code:        000000000001B52A
Plant:                00

CPUs Total:           39
CPUs Configured:      25
CPUs Standby:         0
CPUs Reserved:        14
Capability:           1968
Adjustment 02-way:    242
.
.
.
Adjustment 39-way:    156

LPAR Number:          24
LPAR Characteristics: Shared
LPAR Name:            PETLVS
LPAR Adjustment:      285
LPAR CPUs Total:      2
LPAR CPUs Configured: 2
LPAR CPUs Standby:    0
LPAR CPUs Reserved:   0
LPAR CPUs Dedicated:  0
LPAR CPUs Shared:     2

VM00 Name:            LITHUB
VM00 Control Program: z/VM    5.2.0
VM00 Adjustment:      1000
VM00 CPUs Total:      2
VM00 CPUs Configured: 2
VM00 CPUs Standby:    0
VM00 CPUs Reserved:   0
```

As you can see a lot of valuable system diagnostic information can be ascertained by executing this script. To automate this process, we wrote a script that executes **dbginfo.sh** on each system from our central server. A list of host names is used as input in to the script and contact to the remote systems is done using ssh keys. Once **dbginfo.sh** has completed, **cron** is used to **scp** the file to our central log storage server.

Configuring centralized application core dumps

As with normal log files, it also makes sense to configure user space application core files to be generated in a centralized location. A core dump is a snapshot in time of the state of system memory for a given program. This snapshot may also contain some other useful debugging information and is usually generated under abnormal termination situations. These core files are often requested by software debuggers and can be quite handy in tracking down problems in the field. By default, your system is probably not generating core files or is placing them all over your file system. For our purposes, we chose to create a new directory in /var/log/crash/ to act as our core file repository on each system. To do this, we executed the following commands:

1. We created the new directory:

```
$> mkdir -p /var/log/crash
$> chmod 777 /var/log/crash
```

You can, of course, change the path to some other local directory, as well (possibly even an NFS mount point to a centralized server).

2. We ensured that our systems were capable of producing core files. To check the value of the core file size, execute the following command:

   ```
   $> ulimit -a
   ```

   The **ulimit** utility sets a maximum capacity for files generated by the shell and its forked processes. If your execution returns a string saying unlimited, there is no maximum capacity and generated core files will not be capped in size.

   Should you have a current limit set below your maximum expected core file size, you can increase the size with a variant of the **ulimit** command. For instance you can issue:

   ```
   $> ulimit -c 50000
   ```

3. We needed to tell Linux how we would like the core file names to be written. It certainly makes sense to use a normalized file name convention, and luckily there is an extremely easy interface for doing so.

   To check the current conventions your operating system is using to name core files, simply execute:

   ```
   $> cat /proc/sys/kernel/core_pattern
   ```

   The response as you see it is the format of the core file name to be used. If you simply see `core` it means that core files will be generated in the working directory of the crashed binary, with the non-descriptive name `core`. To specify a more interesting naming convention, echo the format into the /proc/sys/kernel/core_pattern file.

   For example, we did the following:

   ```
   $> echo "/var/log/crash/%h\_%e_%t" >  /proc/sys/kernel/core_pattern
   ```

   The preceding `/var/log/crash/` corresponds to the centralized core file directory we created earlier. (Note that the directory specified in this manner *must* be created in advance. The core dump process will not create it on the fly.) The **%h**, **%e**, and **%t** directives simply indicate to append the host name, executable name, and a time stamp. We chose to use the underscore (_) as our delimiter, but you can select whatever works best for your environment.

To check that the alterations are correct, we began by issuing the following command, which should now display the updated format to be used for the core file name:

```
#> cat /proc/sys/kernel/core_pattern
```

The display properly reflected the change we made, so we used the following procedure to test our configuration. From any directory that is not the centralized core repository that we specified earlier, we executed a binary (for instance, the **top** utility) and ran it in the background using the **&** shell operator. The output shown immediately after the command is the process ID of our background process. (Note that if you have multiple background processes, you will need to look at the number in square brackets to identify which process to recall to the foreground later.)

```
$> top &
[1] 20169
```

Now we will ruthlessly kill that poor, unsuspecting process to simulate what occurs during an actual core dump situation. We do this by issuing the **kill**

command with the abort signal, ABRT, as specified by the appropriate integer, and the process ID displayed for our background execution of the **top** command:

```
kill -6 20169
```

Now that we have delivered the fatal blow, we bring the top instance back to the foreground to observe how it handles the abort situation:

```
$>  fg
top
Aborted (core dumped)
```

Note that the terminal indicates a core file was generated. Using the **ls** command, we see no core file in the current directory. But if we do an **ls** on the /var/log/crash directory, we now see something such as the litstat4_top_1193772520.core entry found on one of our test systems.

You might opt to validate that the core file is readable by executing the gnu debugger on the core file:

```
$> gdb printargs litstat4_top_1193776829.core
```

The output should indicate that a signal 6 was received causing the core file generation to the **top** program.

At this point, if everything looks good, it is time to place these changes into production such that they persist across reboot.

As root, edit the sysctl.conf file:

```
$> vi /etc/sysctl.conf
```

Append the following line, corresponding to our (or your own) customizations:

```
"kernel.core_pattern = /var/log/crash/%h_%e_%t.core"
```

Test once again by issuing the **sysctl -p** command to reload the system control file.

You can ensure that the sysctl configuration is read at boot time by your system initialization scripts through the use of the **chkconfig** tool:

```
$> chkconfig boot.sysctl
boot.sysctl  on
```

If everything checks out, one final test after a reboot should confirm that the setup is operational:

```
litstat4:~ # ulimit -c unlimited

litstat4:~ # top &
[1] 1463

litstat4:~ # kill -6 1463
[1]+  Stopped                 top

litstat4:~ # fg
top
Aborted (core dumped)
```

Chapter 13. Active server monitoring

Typically, active server monitoring requires the installation of one or more agents, or clients, that can provide real time data about a server and its workloads. The theme of consolidation we saw in Chapter 12, "Centralized system logging," on page 133 also applies to active monitoring. In most cases, active management clients can be configured to send data to a centralized console or portal. In fact, a single portal may be used for data streams from z/VM and Linux.

Note that different servers require different levels of active monitoring. It is important to understand that these clients often incur some performance penalty on the server they are monitoring. Although multiple active management clients can be installed, it makes more sense to deploy the minimum amount of active monitoring clients necessary for each deployment. For this test report, we have shown a few of these clients and present our lessons learned during their deployment. This is by no means a comprehensive list but should be viewed as an introduction to the subject matter and a few of the options available. As time permits, we intend to review other management solutions in ongoing phases of our systems management investigations and the associated subsequent test reports.

Setting up Linux guests to cut z/VM monitor records

When running a number of Linux systems as z/VM guests, it can be difficult to collectively gather real time performance data for analysis. By enabling Linux guests to cut z/VM performance records, you have the ability to analyze Linux performance data from the Linux perspective and the z/VM perspective in a single, centralized z/VM tool. This tool is the Performance Toolkit for z/VM. In addition to simplifying the collection of disparate data sets, when Linux guests are configured to write z/VM monitor records a net resource reduction can be observed. By aggregating the data centrally, a single tool may be used for analysis instead of executing computationally expensive monitoring clients on each individual guest. We will demonstrate how to configure your Linux guests to write the necessary monitor records, as well as the use of the z/VM Performance Toolkit to analyze the data.

Starting with SuSE Linux Enterprise Server 9 (SLES 9) and RedHat Enterprise Linux 4 (RHEL 4), Linux systems running as z/VM guests have the ability to cut z/VM *Monitor records using the **appldata_os**, **appldata_mem**, and **appldata_net_sum** modules to report performance information. Examples of the relevant information include CPU usage, memory usage, and network statistics. These performance reporting modules enable Linux to post performance data into the z/VM CP monitoring buffer.

Before a Linux guest can post performance data to the CP monitoring buffer, the guest must have appropriate authorization to do so. This is done by adding the APPLMON option statement to the z/VM directory entry for the guest, as shown in Figure 38 on page 142.

```
USER LITDAT01 99999999 1536M 2048M GZ
   INCLUDE LIN191
   CPU 0
   CPU 1
   IPL CMS PARM AUTOCR
   MACHINE ESA 3
   OPTION APPLMON
```

Figure 38. Example of the OPTION APPLMON statement in the z/VM directory for a Linux guest

The process of implementing Linux to post monitor records to the CP monitoring buffer is slightly different on SuSE and RedHat. We explain the differences in "Enabling monitor records on RedHat Enterprise Linux" and "Enabling monitor records on SuSE Linux Enterprise Server."

Enabling monitor records on RedHat Enterprise Linux

On RedHat Enterprise Linux, you must enable each of the appldata types in /etc/sysctl.conf, as shown in Figure 39, and reboot the system.

```
/etc/sysctl.conf
     #
     # Enable *Monitor records to be cut for z/VM
     appldata.mem = 1
     appldata.os = 1
     appldata.timer= 1
     appldata.net_sum = 1
```

Figure 39. Enabling monitor records on RedHat Enterprise Linux

Enabling monitor records on SuSE Linux Enterprise Server

SuSE systems require the following three steps to enable cutting monitor records:

1. Add the four appldata modules to /etc/sysconfig/kernel, run **mkinitrd** and **zipl**. This will make your changes persistent across IPLs.
   - The **mkinitrd** utility creates a new RAM disk image with the modules specified in the /etc/sysconfig/kernel file.
   - The **zipl** utility places the newly created RAM disk into production. This RAM disk will be loaded at the next IPL.
2. Enable the appldata flags in sysctl.conf.
3. Add `sysctl -p` to /etc/init.d/boot.local.

Additional detail about each of these steps follows:

1. Add the appldata modules to /etc/sysconfig/kernel to the statement MODULES_LOADED_ON_BOOT="". For example:

   ```
   # This variable contains the list of modules to be loaded
   # once the main filesystem is active
   #
   #MODULES_LOADED_ON_BOOT=""
   MODULES_LOADED_ON_BOOT="appldata_mem appldata_net_sum appldata_os"
   ```

   Save the file and run **mkinitrd** followed by **zipl** to write out the new RAM disk images with the new appldata drivers enabled.
2. Add the appldata statements to /etc/sysctl.conf. This step is the same on Suse and RedHat. For example:

   ```
   /etc/sysctl.conf
        #
        # Enable *Monitor records to be cut for z/VM
   ```

```
appldata.mem = 1
appldata.os = 1
appldata.timer= 1
appldata.net_sum = 1
```

3. Add a line to your /etc/init.d/boot/local file containing the line `sysctl -p` and reboot the system. For example:

```
cat boot.local
#! /bin/sh
#
# Copyright (c) 2002 SuSE Linux AG Nuernberg, Germany.  All rights reserved.
#
# Author: Werner Fink <werner>, 1996
#         Burchard Steinbild, 1996
#
# /etc/init.d/boot.local
#
# script with local commands to be executed from init on system startup
#
# Here you should add things, that should happen directly after booting
# before we're going to the first run level.
#
sysctl -p
```

Rebooting the system

After updating the configuration files, a reboot is required to pick up the changes on both SuSE or RedHat. You can verify the Linux system is posting performance data to the CP monitoring buffer by reviewing the data within the z/VM Performance Toolkit.

Using the z/VM Performance Toolkit with Linux

We will now see how to view performance data posted to the CP Monitor from Linux. This topic assumes that the z/VM Performance Toolkit has previously been installed, configured, and is currently running at your site. If you are not familiar with the Performance Toolkit for z/VM or would like information about installation and configuration, see the IBM Redbook *Linux on IBM eServer™ zSeries and S/390: Performance Toolkit for VM*, SG24-6059, available from the IBM Redbooks Web site at www.redbooks.ibm.com/redbooks/pdfs/sg246059.pdf.

From the Performance Toolkit main menu, in the User Data section, select option 29, Linux Systems. This is the option we want to view data that our Linux guests have posted to the CP Monitor. Selecting this menu option will display a Linux panel with four choices, as shown in Figure 40.

```
  Linux screens selection
S Display     Description
. LINUX       RMF PM system selection menu
. LXCPU       Summary CPU activity display
. LXMEM       Summary memory util. & activity display
. LXNETWRK    Summary network activity display
```

Figure 40. The Linux menu, from the main menu of Performance Toolkit for z/VM

The selections on the Linux menu are:

LINUX

The first selection, LINUX, is from an earlier RMF PM (Resource Monitoring Facility - Performance Monitoring) implementation that used an XML request sent to the Linux guest over TCP/IP from the Performance

Toolkit to gather data. For this report, our interest is on the last three choices, LXCPU, LXMEM and LXNETWRK.

LXCPU
The LXCPU panel consolidates the CPU statistics for all Linux guests posting monitoring records in a single view. At a glance, the LXCPU panel displays the number of CPUs defined to the Linux guest, total guest CPU consumption, CPU percentage per user, Kernel/idle processing percentages, as well as information about the total number of processes, runable processes, and processes waiting to run.

LXMEM
The LXMEM panel displays information on total main memory, memory used, buffer memory, swapping utilization, size and rates. The LXMEN is also a consolidated view like LXCPU.

LXNETWRK
The LXNETWRK screen displays the number of network interfaces defined, data transfer rates (as measure in packets and bytes) as well as network error information.

The LXCPU, LXMEM, and LXNETWRK views report statistics about the Linux guests from the Linux image point of view, but what if you want to look at Linux performance information from a z/VM point of view? As it turns out there are a number of options:

- From the main Performance Toolkit menu, option 21, User resource usage, displays CPU utilization for not only the Linux guests, but all guests running on the z/VM system. This allows you to monitor and balance resources used within z/VM. This panel also displays the share value set for the guest.
- Option 11 from the main menu, Channel load, displays OSA utilization which is a factor often overlooked. When adding additional network servers to your environment, the channel load panel is useful for balancing your network traffic loads.
- Option 8, LPAR data, displays the number of CPUs allocated to each LPAR and their CPU utilization. This can be useful to gather and compare data from Linux systems running in an LPAR.

The Performance Toolkit for z/VM has a number of other useful features. For more information, we recommend reading *Linux on IBM eServer zSeries and S/390: Performance Toolkit for VM*, cited above.

IBM Tivoli Monitoring

IBM Tivoli Monitoring (ITM) monitors and manages system and network applications on a variety of platforms and keeps track of the availability and performance of all parts of your enterprise. IBM Tivoli Monitoring provides reports you can use to track trends and troubleshoot problems.

As part of our integration and consolidation with z/OS, wherever possible, we planned to take advantage of the z/OS PET team's existing environment which utilizes the Tivoli OMEGAMON XE suite of products to monitor the environment. All of the data collected is sent to the Tivoli Enterprise™ Management Server (TEMS) running on z/OS and is monitored using the Tivoli Portal Server (TEPS) which is installed on a Windows workstation. To incorporate our monitoring with z/OS PET, we installed the Tivoli Monitoring Agent Version 6.1 on our Linux LPARs.

Installing the Tivoli monitoring agent

We downloaded the code and installed and customized the product. The following is the sequence of prompts and responses for the purpose of showing you what we selected:

```
litstat6:~ # cd /media/mw/Omegamon/
litstat6:/media/mw/Omegamon # ls
.   Deploy        InstallITM  kcirunas.cfg     notices             unix
..  DeployLnk.sh  install.sh  non_ibm_license  silent_install.txt
litstat6:/media/mw/Omegamon # ./install.sh
INSTALL

Enter the name of the IBM Tivoli Monitoring directory
[ default = /opt/IBM/ITM ]:

"/opt/IBM/ITM" does not exist
Try to create it [ y or n; "y" is default ]?

Select one of the following:

1) Install products to the local host.
2) Install products to depot for remote deployment (requires TEMS).
3) Exit install.

Please enter a valid number:  1

Initializing ...
Software Licensing Agreement
1. Czech
2. English
3. French
4. German
5. Italian
6. Polish
7. Portuguese
8. Spanish
9. Turkish

Please enter the number that corresponds to the language
you prefer.

2

Software Licensing Agreement
Press Enter to display the license agreement on your
screen. Please read the agreement carefully before
installing the Program. After reading the agreement, you
will be given the opportunity to accept it or decline it.
If you choose to decline the agreement, installation will
not be completed and you will not be able to use the
Program.

International Program License Agreement

Part 1 - General Terms

BY DOWNLOADING, INSTALLING, COPYING, ACCESSING, OR USING
THE PROGRAM YOU AGREE TO THE TERMS OF THIS AGREEMENT. IF
YOU ARE ACCEPTING THESE TERMS ON BEHALF OF ANOTHER PERSON
OR A COMPANY OR OTHER LEGAL ENTITY, YOU REPRESENT AND
WARRANT THAT YOU HAVE FULL AUTHORITY TO BIND THAT PERSON,
COMPANY, OR LEGAL ENTITY TO THESE TERMS. IF YOU DO NOT
AGREE TO THESE TERMS,
```

```
- DO NOT DOWNLOAD, INSTALL, COPY, ACCESS, OR USE THE
PROGRAM; AND

- PROMPTLY RETURN THE PROGRAM AND PROOF OF ENTITLEMENT TO

Press Enter to continue viewing the license agreement, or,
Enter "1" to accept the agreement, "2" to decline it or
"99" to go back to the previous screen.

1

Preparing to install the IBM Global Security Kit (GSkit)
Preparing packages for installation...
gsk7bas-7.0-3.18
Preparing packages for installation...
gsk7bas64-7.0-3.18
 Will enable automatic agent initiation after reboot.

Enter a 32-character encryption key, or just press Enter to use the default
        Default = IBMTivoliMonitoringEncryptionKey
....+....1....+....2....+....3..

enter

GSkit encryption key has been set.
Key File directory: /opt/IBM/ITM/keyfiles

 Product packages are available in /media/mw/Omegamon/unix

Product packages are available for the following operating systems and component
support categories:

 1) Linux S390 R2.4 (32 bit)
 2) Linux S390 R2.4 (64 bit)
 3) Linux S390 R2.4 GCC 2.9.5 (64 bit)
 4) Linux S390 R2.6 (32 bit)
 5) Linux S390 R2.6 (64 bit)
 6) Linux S390 R2.6 GCC 2.9.5 (32 bit)
 7) Linux S390 R2.6 GCC 2.9.5 (64 bit)
 8) Tivoli Enterprise Portal Browser Client support
 9) Tivoli Enterprise Portal Server support

Type the number for the OS or component support category you want, or type "q" to
quit selection
[ number "5" or "Linux S390 R2.6 (64 bit)" is default ]:  5

You selected number "5" or "Linux S390 R2.6 (64 bit)"

Is the operating system or component support correct [ y or n; "y" is default ]?

The following products are available for installation:

 1) IBM Eclipse Help Server  V06.10.05.01
 2) Monitoring Agent for Linux OS  V06.10.05.01
 3) Monitoring Agent for UNIX Logs  V06.10.05.01
 4) Summarization and Pruning Agent  V06.10.05.01
 5) Tivoli Enterprise Monitoring Server  V06.10.05.01
 6) Tivoli Enterprise Services User Interface  V06.10.05.01
 7) Universal Agent  V06.10.05.01
 8) Warehouse Proxy  V06.10.05.01
 9) all of the above

Type the numbers for the products you want to install, or type "q" to
quit selection.
If you enter more than one number, separate the numbers by a comma or
```

```
a space.

Type your selections here:  2

The following products will be installed:

  Monitoring Agent for Linux OS  V06.10.05.01

Are your selections correct [ y or n; "y" is default ]?

 ... installing "Monitoring Agent for Linux OS  V06.10.05.01 for Linux S390
R2.6 (64 bit)";  please wait.

 => installed "Monitoring Agent for Linux OS  V06.10.05.01 for Linux S390
R2.6 (64 bit)."
 ... Initializing database for Monitoring Agent for Linux OS  V06.10.05.01 for
Linux S390
 R2.6 (64 bit).

Do you want to install additional products or product support packages
[ y or n; "n" is default ]? n

 ... postprocessing; please wait.

 ... finished postprocessing.

 Installation step complete.

 As a reminder, you should install product support on each of your TEM
 servers for any agents you have just installed. This is done via the
 "[ITM home]/bin/itmcmd support" command on your TEM servers.

 You may now configure any locally installed IBM Tivoli Monitoring product via
 the "/opt/IBM/ITM/bin/itmcmd config" command.

/opt/IBM/ITM/bin
```

Configuring the Tivoli monitoring agent

The **itmcmd config** command is used to configure the execution environment for the agent. This tells the agent the operating system, network protocols, and where to send the data to be monitored. Various options for secondary network protocols and management servers were offered but we basically went with the defaults. In our case, the Tivoli Enterprise Management Server was running on z/OS system JCOEIP. In the following example, we used line commands to configure but the GUI can also be used by issuing the **./itmcmd manage** command.

```
litstat6:/opt/IBM/ITM/bin # ./itmcmd config -A lz
(-A = Agent, lz = product code for Linux)

Agent configuration started...

Will this agent connect to a TEMS? [YES or NO] (Default is: YES): YES
TEMS Host Name (Default is: litstat6): JC0EIP

Network Protocol [ip, sna, ip.pipe or ip.spipe] (Default is: ip.pipe): ip

     Now choose the next protocol from one of these:
     - sna
     - ip.pipe
     - ip.spipe
     - none
Network Protocol 2 (Default is: none):
```

```
IP Port Number (Default is: 1918): 1918

Configure connection for a secondary TEMS? [YES or NO] (Default is: NO): NO
Enter Optional Primary Network Name or "none" (Default is: none): none
Agent configuration completed...
```

With the agents installed, configured, and running on all the LPARs, we went to the Tivoli Enterprise Portal Server to observe our Linux systems but they weren't there! Further investigation revealed the Tivoli Enterprise Management Server on z/OS needed to have the application support for Linux added.

The following outlines how to add application support for Linux to the Tivoli Enterprise Management Server on z/OS:

1. From the Windows machine with the TEP installed, using Manage Tivoli Services, right click the TEPS, then go to **Advanced > Utilities > FTP Catalog and Attribute files**.
2. With the product support already installed, you should select both files for Linux support:

   Attribute data for z/VM and Linux Support

   Catalog Data for z/VM and Linux Support
3. When prompted, enter the host information for the z/OS TEMS.

Upon completion of these steps, you should see a popup saying the transfer was a success.

With the application support now added, the TEMS can be seeded. The following procedure adds application support to the monitoring server from the Windows or Linux computer that hosts the portal server.

1. If you have not already configured and started the hub monitoring server, do so now.
2. If you have not already done so, install the application support data from the *IBM Tivoli OMEGAMON Data Files for z/OS* CD-ROM on the computer that hosts the portal server.
   a. Log onto the Windows computer under a user ID with Administrator authority.
   b. Insert the *IBM Tivoli OMEGAMON Data Files for z/OS* CD-ROM into the CD drive. Installation begins automatically. If the InstallShield wizard does not start, go to the WINDOWS directory on the CD-ROM and run setup.exe. If setup.exe initialization fails, you do not have enough disk space to decompress the setup files.
   c. Follow the prompts to install the application support data files.
3. On the same Windows computer, select **Start > Programs > IBM Tivoli Monitoring > Manage Tivoli Monitoring Services**.
4. From the **Actions** menu in the Manage Tivoli Monitoring Services window, select **Advanced > Add TEMS Application Support**.
5. In the Add TEMS Application Support window, select **On a different computer** and click **OK**.
6. When you are prompted to ensure that the monitoring server is configured and running, click **OK**.
7. On the Non-resident TEMS Connection window, provide the TEMS Node ID, which you can find as the value of the CMS_NODEID variable in this location: &RHILEV...&RTE...RKANPAR(KDSENV)
8. Select the appropriate communications protocol and click **OK**.

9. In the next window, provide any values required by your communications protocol. (For example, if your protocol is TCP/IP, the window prompts you for the TCP/IP host name and port number of the monitoring server to receive application support.)
10. On the Select Product to Add TEMS Application Support window, select the products whose SQL data (predefined situations, policies, and take action commands) you want to add to the monitoring server configuration, and click **OK**.
11. Upon completion (this might take several minutes), the Add TEMS Application Support Operation Complete window gives you information about the status and data location. If desired, click **Save As** to save the information in a text file. Click **Close** to close the window.

Now with all the required support applied, we can view our Linux LPARs on the TEP, as shown in Figure 41. (Linux systems litrwas3, litsha23, litstat5, litstat6, and litswas3.)

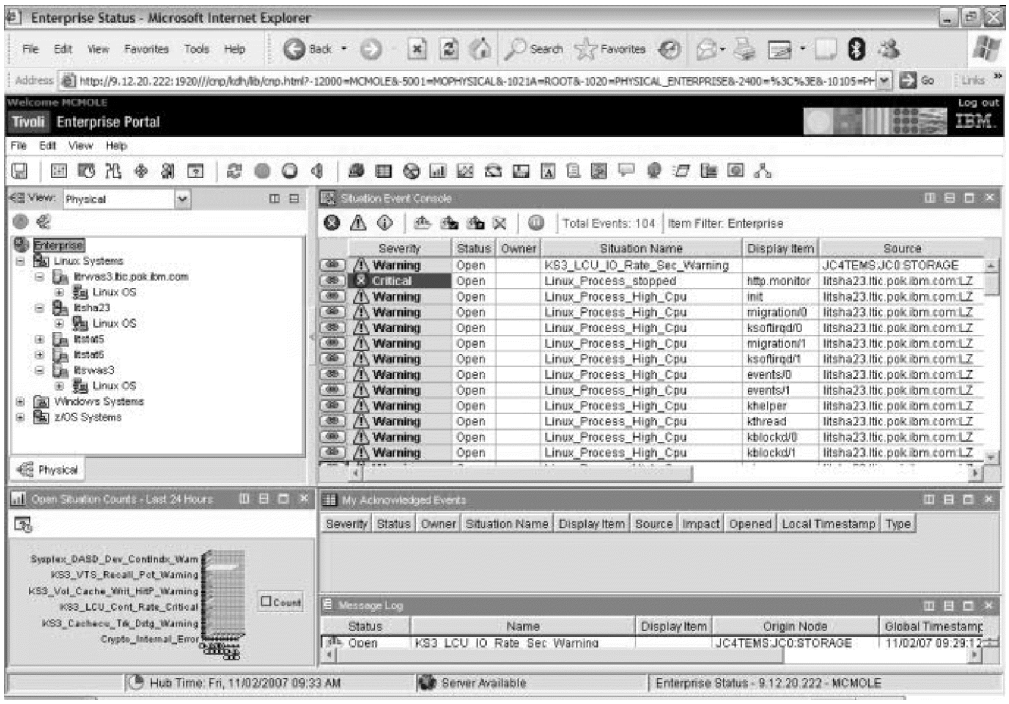


Figure 41. Linux systems displayed on TEP

In Figure 42 on page 150, we've selected Linux system litrwas3 to show the workspaces that are available to be viewed. They include capacity usage information, disk usage, file information, network, process, system information and users.

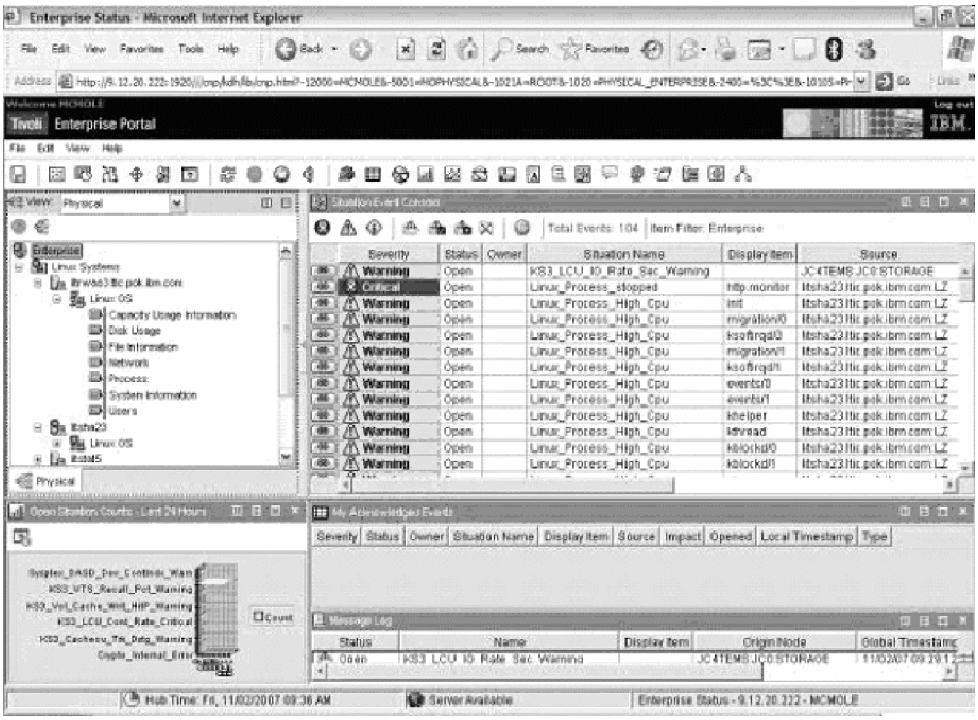


Figure 42. Available workspaces to view on Linux system litrwas3

In the next phase of our project, we plan to install IBM Tivoli OMEGAMON XE on z/VM and Linux, which will enable us to monitor our remaining Linux systems.

Event management using IBM Tivoli Enterprise Console

IBM Tivoli Enterprise Console® is a rule-based event management application that integrates system, network, database, and application management to help ensure the optimal availability of the IT resources in an enterprise. Our intention is to integrate this tool in to our environment to monitor and react to Simple Network Management Protocol (SNMP) alerts. First a bit about the product features.

The Tivoli Enterprise Console product:

- Provides a centralized, global view of your computing enterprise
- Collects, processes, and automatically responds to common management events, such as a database server that is not responding, a lost network connection, or a successfully completed batch processing job
- Acts as a central collection point for alarms and events from a variety of sources, including those from other Tivoli software applications, Tivoli partner applications, custom applications, network management platforms, and relational database systems

The Tivoli Enterprise Console product helps you effectively process the high volume of events in an IT environment by:

- Prioritizing events by their level of importance
- Filtering redundant or low-priority events
- Correlating events with other events from different sources
- Determining who should view and process specific events

- Initiating automatic corrective actions, when appropriate, such as escalation, notification and the opening of trouble tickets
- Identifying hosts and automatically grouping events from the hosts that are in maintenance mode in a predefined event group

Components of Tivoli Enterprise Console

Following are the Tivoli Enterprise Console components, which are listed in the order in which an event flows from its source to the operator:

- Adapter Configuration Facility
- Event adapter
- Tivoli Event Integration Facility
- Tivoli Enterprise Console gateway
- Event server
- Event database
- User interface server
- Event console

Installing Tivoli Management Framework V4.1.1 on Linux for System z

We installed and configured IBM Tivoli Enterprise Console V3.9, Patch 06 for event management. In order to use Tivoli Enterprise Console (TEC), we first had to install Tivoli Management Framework (TMF) V4.1.1. The Tivoli Desktop product is used to install Tivoli products, update, start, stop, check the status of the Event Server and do other configuration tasks. It's not supported on Linux for System z, so we chose to install the Tivoli Desktop on a Windows 2000 system. In this report we will show you some of the obstacles we encountered during our setup, how they were resolved, and what we monitored.

The first product CD-ROM (CD 1), of two, contained TMF V4.1.1. This install required a patch (4.1.1-TMF-0001) to be applied to the content of product CD. We applied the patch as follows:

1. Copied the content of CD 1 to our local file systems
2. Extracted the files from patch file, 4.1.1-TMF-0001.tar
3. Followed the README file for patch 4.1.1-TMF-0001 and replaced files

We created an installation directory and ran the WPREINST.SH script which is located in the TMF directory. This process copies some files and prompts for another command to run:

```
LITSTEC:/usr/local/Tivoli/download/TMF # ./WPREINST.SH
to install, type ./wserver -c .
```

Continuing with the directions, we ran the following command:

```
LITSTEC:/usr/local/Tivoli/download/TMF # ./wserver -c .
./wserver: line 178: /usr/local/Tivoli/download/TMF/bin/linux-s390/wserver_gui:
No such file or directory
```

We received an error because TMF on Linux for System z does not support the GUI interface, so we had to install TMF manually. To do this, we set the DOGUI environment variable to "no" using the following commands:

```
DOGUI=NO
EXPORT DOGUI
```

Ran the command again:

```
LITSTEC:/usr/local/Tivoli/download/TMF # ./wserver -c
BIN=/usr/local/Tivoli/bin \
LIB=/usr/local/Tivoli/lib \
ALIDB=/usr/local/Tivoli/db \
MAN=/usr/local/Tivoli/man \
APPD=/usr/local/Tivoli/X11/app-defaults \
CAT=/usr/local/Tivoli/msg_cat \
AutoStart=1 \
SetPort=1 \
CreatePaths=1
```

The command still failed as the installation copied files into a temporary directory and created all the links but this temporary directory was in the same directory structure as the TMF install directory. Instead of writing the full directory path for commands, it used the relative path which caused the installation to fail. To avoid this we had to create a temporary directory on /tmp (Install_Tiv) and then run command:

```
LITSTEC:~/Install_Tiv # /usr/local/Tivoli/download/TMF/WPREINST.SH
LITSTEC:~/Install_Tiv # ./wserver -c /usr/local/Tivoli/download/TMF BIN=/us
al/Tivoli/bin \
>  LIB=/usr/local/Tivoli/lib \
>  ALIDB=/usr/local/Tivoli/db \
>  MAN=/usr/local/Tivoli/man \
>  APPD=/usr/local/Tivoli/X11/app-defaults \
>  CAT=/usr/local/Tivoli/msg_cat \
>  AutoStart=1 \
>  SetPort=1 \
>  CreatePaths=1

Using command line style installation...
Unless you cancel, the following operations will be executed:
.
.
   Distributing architecture specific 371_Libraries --> LITSTEC
. (repeated lines eliminate for brevity)

.... Completed.

Registering installation information...Finished.
```

After the installation was completed, checked the TMR status by running following commands:

```
LITSTEC:/etc/Tivoli # . ./etc/Tivoli//setup_env.sh
LITSTEC:/etc/Tivoli # odadmin odlist
Region              Disp Flags Port IPaddr  Hostname(s)
1436291848          1 ct- 94 192.168.71.16 LITSTEC.pdl.pok.ibm.com
```

Installing Tivoli Desktop on a Windows 2000 system

IBM Tivoli Desktop install code was on the second CD-ROM (CD 2). We ran the setup.exe from CD 2.

Now to connect to our TMF running on Linux for System z: On the Windows system: **Start > Programs > Tivoli > Tivoli Desktop** and enter the TMF server host name, user ID, and password. Upon connection to TMF, you will see Tivoli Desktop, as shown in Figure 43 on page 153.

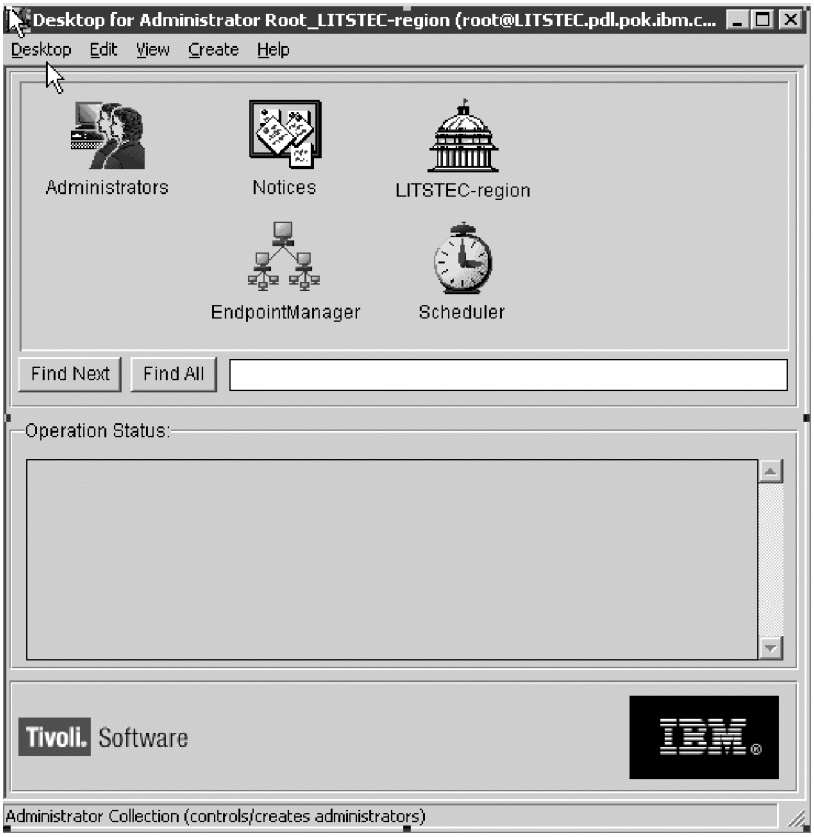


Figure 43. Tivoli Desktop

After installing TMF, we installed Tivoli Framework Patches 4.1.1-TMF-0090 and 4.1.1-LCF-0049 from the Tivoli Desktop:

1. Click **Desktop > Install > Install Patch...**
2. Click the **Select Media** tab and set the PATH to where you have your patch file's *.IND files.

Installing DB2 server

The Tivoli Enterprise Console product requires an external relational database management system (RDBMS) to store the large amounts of event data that it receives. Following the IBM DB2 Installation document, we installed DB2 Server V8.2 as our RDBMS server and created db2inst1 as the DB2 instance owner for our TEC database.

Creating the TEC database and RIM object

The TEC GUI Installer does provide support to create scripts that will create the TEC database, tables, DB2 schemas, and Rim object. However, the TEC does not provide GUI support on Linux for System z so we created all of our scripts on a System x™ server running Linux.

As stated in the Tivoli TEC documentation, the TEC 3.9 installer does not work with Tivoli Frame Work 4.1.1. We used the installer from the TEC Patch 006, file: 3.9.0-TIV-TEC-FP0006-INSTALLASSISTANT.tar.

We extracted file 3.9.0-TIV-TEC-FP0006-INSTALLASSISTANT.tar and ran **tec_install.sh**:

```
[root@jibe06 tivoli]# tar -xvf 3.9.0-TIV-TEC-FP0006-INSTALLASSISTANT.tar
[root@jibe06 tivoli]# ls
3.9.0-TIV-TEC-FP0006-INSTALLASSISTANT.tar  INSTALLASSISTANT  tec
cd INSTALLASSISTANT
[root@jibe06 INSTALLASSISTANT]# ./tec_install.sh
```

This will prompt for the location of the temporary file. The default location is /tmp but we used /home/tivoli/tec. Accepted the license agreement, selected **Configure TEC database**, selected **Custom install**, entered `/home/tivoli/tec` for the script file directory and selected **Generate only**. We selected the generate only option because we will copy the script files over to the same Linux system that is running DB2 on System z.

We then copied these script files to our Linux for System z system, LITSTEC, in directory /home/tivoli/tec. The DB2 instance must have read access to these.

Next, we created the DB2 database, tables and schema. Login to LITSTEC with the DB2 instance owner ID (db2inst1) and ran **wdbconfig.sh** from the /home/tivoli/tec directory, passing `cr` as the parameter:

```
db2inst1@LITSTEC:/home/tivoli/tec> ./wdbconfig.sh cr
```

The following are selected lines of output from this command:

```
Vendor is db2.

Do you have a client or server connection to the database on the machine where
his script is being run?
Type '1' for server or '2' for client

1
db2 CREATE DATABASE tec USING CODESET UTF-8 TERRITORY US
DB20000I  The CREATE DATABASE command completed successfully.
db2 connect to tec

 Database Connection Information

 Database server        = DB2/LINUX390 8.2.0
 SQL authorization ID   = DB2INST1
 Local database alias   = TEC

.
.
.

DB20000I  The SQL command completed successfully.

db2 connect reset
DB20000I  The SQL command completed successfully.
db2inst1@LITSTEC:/home/tivoli/tec>
```

We did not select the RIM object to be created while creating database scripts, so we issued the following command:

```
LITSTEC:~ # wcrtrim -v DB2 -h LITSTEC -d tec -u db2inst1 -H /opt/IBM/db2/V8.1
     -s LITSTEC -I /home/db2inst1 -t db2inst1 tec
```

Now test the RIM object by running the **wrimtest –l tec** command (enter `x` to release session):

```
LITSTEC:~ # wrimtest -l tec
Resource Type  : RIM
Resource Label : tec
Host Name      : LITSTEC
```

```
User Name      : db2inst1
Vendor         : DB2
Database       : tec
Database Home  : /opt/IBM/db2/V8.1
Server ID      : LITSTEC
Instance Home  : /home/db2inst1
Instance Name  : db2inst1
Opening Regular Session...Session Opened
RIM : Enter Option >x
Releasing session
```

Installing Tivoli Enterprise Console

We used the Tivoli Desktop to install other Tivoli TEC products. The general approach is:

1. Click **Desktop > install > install Product** will pop up another window.
2. Click on **Set media** and select the location where the TEC 3.9 product's *.IND files are located.
3. Select the TEC component product you want to install.

From the Tivoli Desktop, we installed the following products and the TEC Patch 006 for each component:

- JRE
- TEC Server
- UI
- Adaptor Configuration Facility
- Sample Event Information 3.9

The first problem we hit was an error, shown in Figure 44 on page 156, that we received while installing TEC Server.

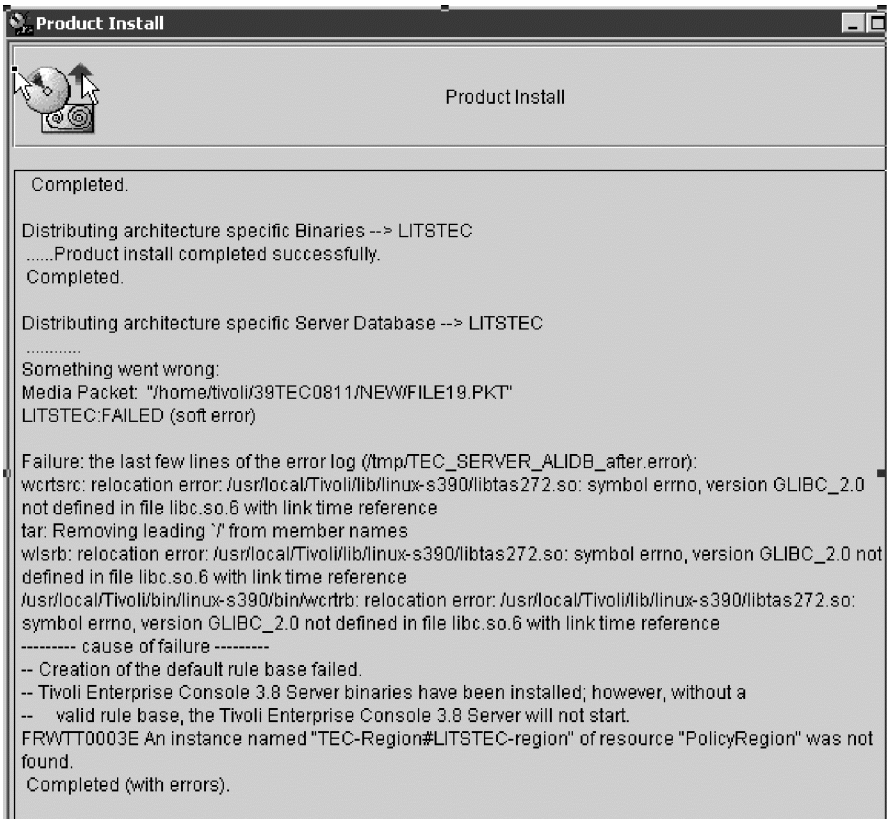


Figure 44. TEC Server installation error

This was a known kernel problem that was resolved with patch 4.1.1-TMF-0001. We had this patch applied but we didn't have the environment variable LD_ASSUME_KERNEL properly set. Per the README file contained in the patch file, we set the environment variable as follows:

```
LITSTEC:/tmp # odadmin environ get > /tmp/env.get

LITSTEC:/tmp # echo LD_ASSUME_KERNEL=2.4.1 >> /tmp/env.get
LITSTEC:/tmp # odadmin environ set < /tmp/env.get
LITSTEC:/tmp # odadmin environ get
BIM_PROLOG_DIR=/usr/local/Tivoli/bin/linux-s390/TME/TEC
PATH=/bin:/usr/bin
SHLIB_PATH=
NLSPATH=/usr/local/Tivoli/msg_cat/%L/%N.cat
TISDIR=/usr/local/Tivoli/bin/linux-s390/../generic
TEC_BIN_DIR=/usr/local/Tivoli/bin/linux-s390/TME/TEC
LD_ASSUME_KERNEL=2.4.1
```

We recycled TMF server using the **odadmin reexec all** command and then reinstalled TEC.

The second problem we hit was an error, shown in Figure 45 on page 157, that we received while installing UI.

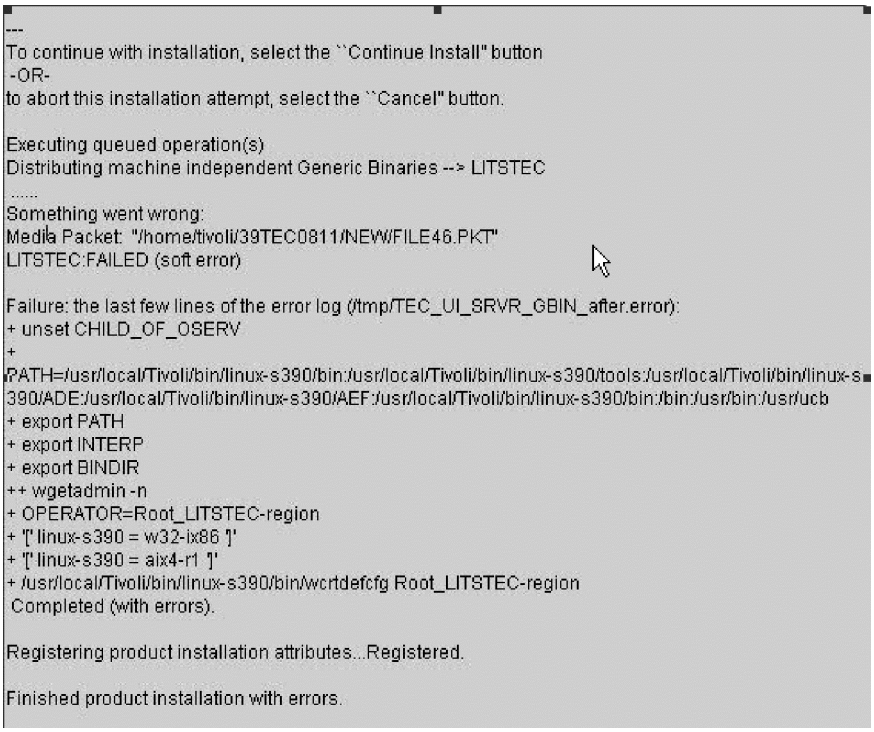


Figure 45. UI installation error

Our investigation revealed this was a known problem and that we should proceed with installing patch 006, which installed successfully and took care of the problem.

Installing TEC console

The TEC Console is not supported on Linux on System z, so we chose to install it on a Windows machine. The updated console package is shipped with TEC Patch 006. We decompressed the file 3.9.0-TIV-TEC-FP0006-NON-TME-W32-IX86.tar.gz on Linux, copied the tar file to our Windows machine, and used the Tivoli command to explode and run the setup.exe.

We started TEC Console, as follows: Clicked **Start > Program > TEC** and entered the host name, user ID, and password. We then saw our TEC Console as shown in Figure 46 on page 158.

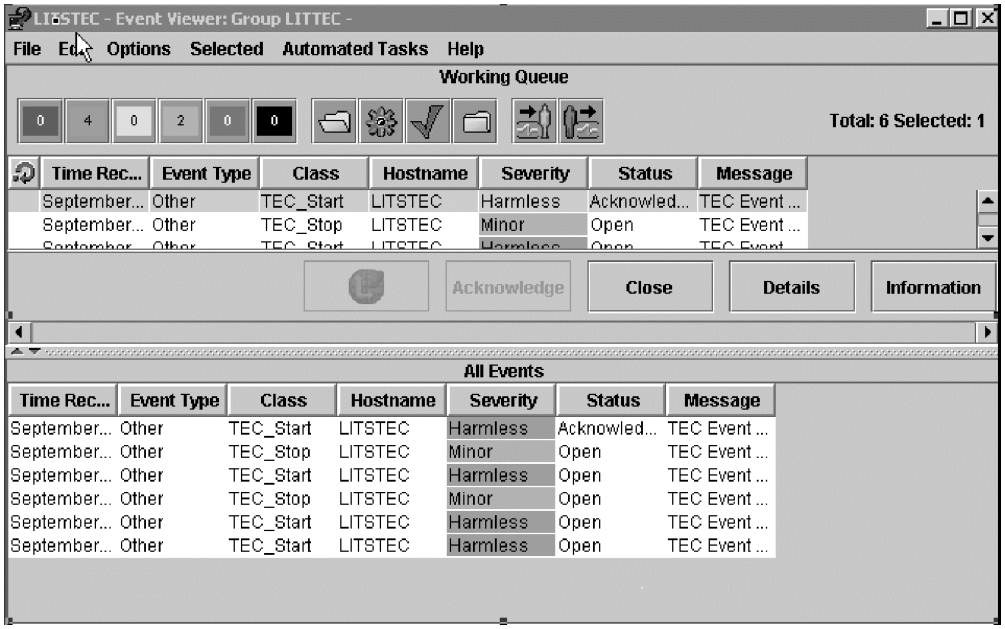


Figure 46. TEC Console display

This completes the Tivoli TMF and TEC Server installations. We logged in to the Tivoli Desktop and recycled the Event Server. This created TEC events and were displayed on our TEC Console, as shown in Figure 44 on page 156.

We then ran the **wtdumpr1** command to see the event in our database:

```
LITSTEC:~ # wtdumprl
1~1~65537~1189180337(Sep 07 11:52:17 2007)
### EVENT ###
TEC_Start;source=TEC;msg="TEC Event Server initialized";hostname=LITSTEC;END

### END EVENT ###
PROCESSED
```

This confirmed that our TEC Server, TEC Console, UI, RIM object, and DB2 were all working fine.

The following is a list of the products and patches that were installed in our environment:

- Products:
  - IBM Tivoli Management Framework V4.1.1
  - IBM Tivoli Enterprise Console JRE 3.9
  - IBM Tivoli Enterprise Console Server 3.9
  - IBM Tivoli Enterprise Console User Interface Server 3.9
  - IBM Tivoli Enterprise Console Adaptor Configuration Facility 3.9
  - IBM Tivoli Enterprise Console Sample Event Information 3.9
- Patches:
  - 3.9.0 Tivoli Enterprise Console Server Fix Pack 6
  - 3.9.0 Tivoli Enterprise Console User Interface Server Fix Pack 6
  - 3.9.0 Tivoli Enterprise Console JRE Fix Pack 6
  - 3.9.0 Tivoli Enterprise Console ACF Fix Pack 6
  - 3.9.0 Tivoli Enterprise Console Sample Event Information Fix Pack 6
  - Tivoli Framework Patch 4.1.1-TMF-0090
  - Tivoli Framework Patch 4.1.1-LCF-0049

- On the Windows 2000 server:
  - IBM Tivoli Enterprise Console Console 3.9
  - IBM Tivoli Desktop V4.1.1

Creating an endpoint node from Tivoli Desktop

A Tivoli environment can include the following types of clients:

- Managed nodes
- Endpoints

A *managed node* runs the full Tivoli Management Framework software and can perform the same security and communication functions performed by the Tivoli server. The Tivoli server is the machine from which system administrators manage other systems in the network. A managed node maintains a client database, which is significantly smaller than the object database on the Tivoli server.

An *endpoint* is the most common type of machine in most Tivoli Management Framework installations. This machine is not used to perform day-to-day management operations. Instead, it is one of the many machines a system administrator must manage, usually from a managed node. An endpoint runs a very small amount of Tivoli software and does not maintain a database.

A Tivoli Managed Region (TMR) communicates to End Point Nodes or Managed Nodes using a Gateway, so we created a Gateway from the Tivoli Desktop:

1. From Tivoli Desktop, right-click on **EndpointManager > create Gateway**
2. Enter Name (LITSTEC-GW),
3. Click on the **Managed Nodes...** tab, select the node, in our case, LITSTEC.

Then we created an endpoint on our LITRWAS4 system using following command:

Note: IBM TMF uses **ssh** or **rexec** to distribute code to the end point nodes. We specified **-j** on the command which indicated we were going to use **ssh**. For **ssh** to work, we had to create a host key on the target end point system. We issued **ssh** to our LITRWAS4 system, logged in as root, and a key was created in the known host file. Without the key, your installation will fail.

```
LITSTEC:~ # winstlcf -j -g 192.168.71.16+9494 -n litrwas4-ep 192.168.71.138

Trying 192.168.71.138...
password for root:

locating files in /usr/local/Tivoli/bin/lcf_bundle.41100...
locating files in /usr/local/Tivoli/bin/lcf_bundle...

Ready to copy files to host 192.168.71.138:
  destination: 192.168.71.138:/opt/Tivoli/lcf
       source: LITSTEC:/usr/local/Tivoli/bin/lcf_bundle.41100
        files:
               generic/lcfd.sh
               generic/epinst.sh
               generic/as.sh
               generic/lcf_env.sh
               generic/lcf_env.csh
               generic/lcf_env.cmd
               generic/lcf.inv
               bin/linux-s390/mrt/lcfd
               lib/linux-s390/libcpl272.so
```

```
                lib/linux-s390/libdes272.so
                lib/linux-s390/libmd2ep272.so
                lib/linux-s390/libguid272.so
                lib/linux-s390/libmrt272.so
                lib/linux-s390/libtis272.so
                lib/linux-s390/libtos.so
                lib/linux-s390/libtthred.so

  Continue? [yYna?]y
Tivoli Light Client Framework starting on 192.168.71.138
Sep 10 11:04:53 1 lcfd Command line argv[0]='/opt/Tivoli/lcf/bin/linux-s390/mrt/lcfd'
Sep 10 11:04:53 1 lcfd Command line argv[1]='-Dlcs.login_interfaces=192.168.71.16+9494'
Sep 10 11:04:53 1 lcfd Command line argv[2]='-n'
Sep 10 11:04:53 1 lcfd Command line argv[3]='litrwas4-ep'
Sep 10 11:04:53 1 lcfd Command line argv[4]='-Dlib_dir=/opt/Tivoli/lcf/lib/linux-s390'
Sep 10 11:04:53 1 lcfd Command line argv[5]='-Dload_dir=/opt/Tivoli/lcf/bin/linux-s390/mrt'
Sep 10 11:04:53 1 lcfd Command line argv[6]='-C/opt/Tivoli/lcf/dat/1'
Sep 10 11:04:53 1 lcfd Command line argv[7]='-Dlcs.machine_name=192.168.71.138'
Sep 10 11:04:53 1 lcfd Command line argv[8]='-Dlcs.login_interfaces=192.168.71.16+9494'
Sep 10 11:04:53 1 lcfd Command line argv[9]='-n'
Sep 10 11:04:53 1 lcfd Command line argv[10]='litrwas4-ep'
Sep 10 11:04:53 1 lcfd Starting Unix daemon
Performing auto start configuration
Tivoli LCF daemon master autostart file is /etc/init.d/Tivoli_lcfd1.
Done
```

We verified the endpoint by running the following commands:

```
LITSTEC:~ # wep ls
G     1436291848.1.712  LITSTEC-GW
1436291848.2.522+#TMF_Endpoint::Endpoint# litrwas4-ep

LITSTEC:~ # wepstatus litrwas4-ep

Endpoint Label      : litrwas4-ep
Dispatcher Number   : 2
Gateway Label       : LITSTEC-GW
Interp Type         : linux-s390
Version             : 41149
IP Address          : 192.168.71.138+9495
Current Status      : connected
Status error code(s):
Last Login Time     : 2007/09/10-11:04:54 [Passed]
Last Logout Time    : NOT_SET
Last Migration Time: 2007/09/10-11:04:53 [Passed]
Last Downcall Time: 2007/09/10-11:04:54 [Passed]
Last Upcall Time    : NOT_SET
Last EP check Time: NOT_SET

LITSTEC:~ #
```

Installing adaptors on the TME system

An adaptor is a process that monitors resources so that they can be managed. These monitored resources are called sources. A source is an application (for example, a database) or system resource (for example, an NFS server). When an adapter detects an event generated from a source (generally called a raw event), it formats the event and sends it to the event server. The event server then further processes the event.

On our TME® systems, we installed the adaptor from the Tivoli Desktop. We created a Profile Manager, added a subscriber to our profile manager, created a profile for the adaptor, dragged the adapter profile and dropped it on the subscriber node:

1. Created a profile manager: Double-clicked on TEC-Region, then **Create > ProfileManager** (we entered LITSTEC_PM in name field).
2. Added a subscriber: Double-clicked on the LITSTEC_PM Profile Manager which brought up the Profile Manager Window. Then, right-clicked on **Profile Manager > Subscriber...**, selected the node we wanted to subscribe, and clicked **OK**.
3. Created a profile for the adaptor: From the Profile Manager Window, right-clicked on **Create > Profile**, entered the logfile as the name of our profile.
4. Edited the profile by right-clicking on the logfile profile icon, then **Edit Profile**. We entered `LOG_WASR4` as the name and clicked on the **Add Entry** tab. Selected **tecad_logfile_linux-s390 adaptors** from the menu.
5. From the Profile Manager Window, we dragged the profile and dropped it on the endpoint, which installed the adaptor.

We installed the LOGFILE and SNMP adaptors on our LITRWAS4 system.

Installing adaptors on the non-TME systems

The Tivoli Enterprise Console Patch 006 package contained all of the adaptors as a full, installable package. The latest patch for TEC was 3.9.0-TIV-TEC-FP0006. The adaptors for the NON-TME LOGFILE and SNMP were in 3.9.0-TIV-TEC-FP0006-NON_TME-LINUX-S390.tar file.

To install the adaptors, we began by extracting the package file to get to LOGFILE.TAR and SNMP.TAR file:

```
/opt/tivoli/adaptor # tar -xvf 3.9.0-TIV-TEC-FP0006-NON_TME-LINUX-S390.tar
NON_TME/LOGFILE/LINUX-S390/
NON_TME/LOGFILE/LINUX-S390/LOGFILE.TAR
NON_TME/SNMP/LINUX-S390/
NON_TME/SNMP/LINUX-S390/SNMP.TAR
```

We did the following to install the LOGFILE adaptor:

1. Extracted the file /opt/tivoli/adaptor/NON_TME/LOGFILE/LINUX-S390/LOGFILE.TAR into the installation directory (directory of your choice)

   ```
   export TECADHOME=/opt/tivoli/adaptor/LOGFILE
   ```

2. Ran the initialization script and responded to the prompts for the event server, port number, and auto start option

   ```
   $TECADHOME/bin/tecad_logfile.cfg
   ```

Commands to start and stop the LOGFILE adaptor are:

```
$TECADHOME/bin/init.tecad_logfile start
$TECADHOME/bin/init.tecad_logfile stop
```

We did the following to install the SNMP adaptor:

1. Extracted the file /opt/tivoli/adaptor/NON_TME/LOGFILE/LINUX-S390/SNMP.TAR into the installation directory (directory of your choice)

   ```
   export TECADHOME=/opt/tivoli/adaptor/SNMP
   ```

2. Ran the initialization script and responded to the prompts for the event server, port number, and auto start option

   ```
   $TECADHOME/bin/tecad_snmp.cfg
   ```

Commands to start and stop the SNMP adaptor are:

```
$TECADHOME/bin/init.tecad_snmp start
$TECADHOME/bin/init.tecad_snmp stop
```

To verify that the adaptors are functioning properly, we manually sent over some events by running the following commands:

```
litstat1:/opt/tivoli/adaptor/LOGFILE/bin # ./postzmsg -S 192.168.71.16 -r WARNING -m "Su Failure" Su_Failure LOGFILE

litstat1:/opt/tivoli/adaptor/SNMP/bin # ./postzmsg -S 192.168.71.16 -r WARNING -m "SNMP_Trap" SNMP_Trap SNMP

litstat1:/opt/tivoli/adaptor/SNMP/bin # ./postzmsg -S 192.168.71.16 -r WARNING -m "Cold Start" Cold_Start SNMP
```

The first display on the TEC console will be the Summary Chart View, as shown in Figure 47.

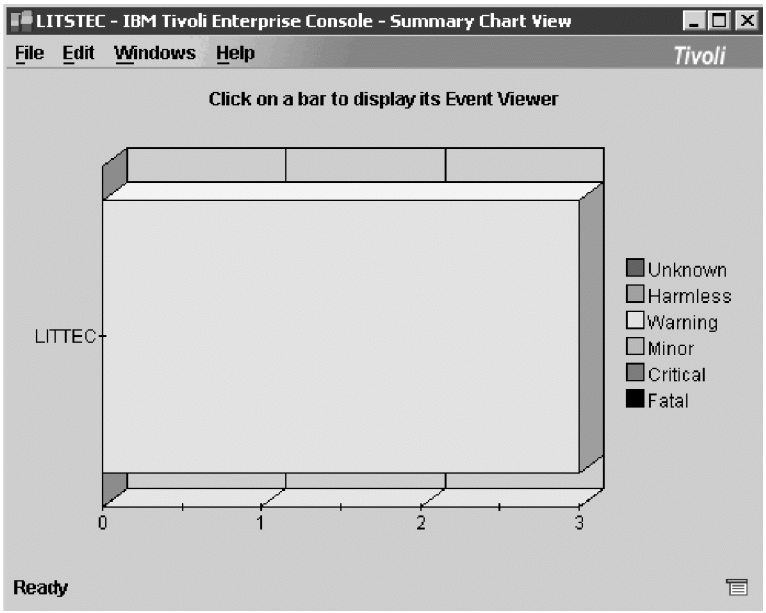


Figure 47. TEC Summary Chart view

Click on any area of the Summary display to bring up the Event Viewer. Here you can view all the details about an event, as shown in Figure 48 on page 163.

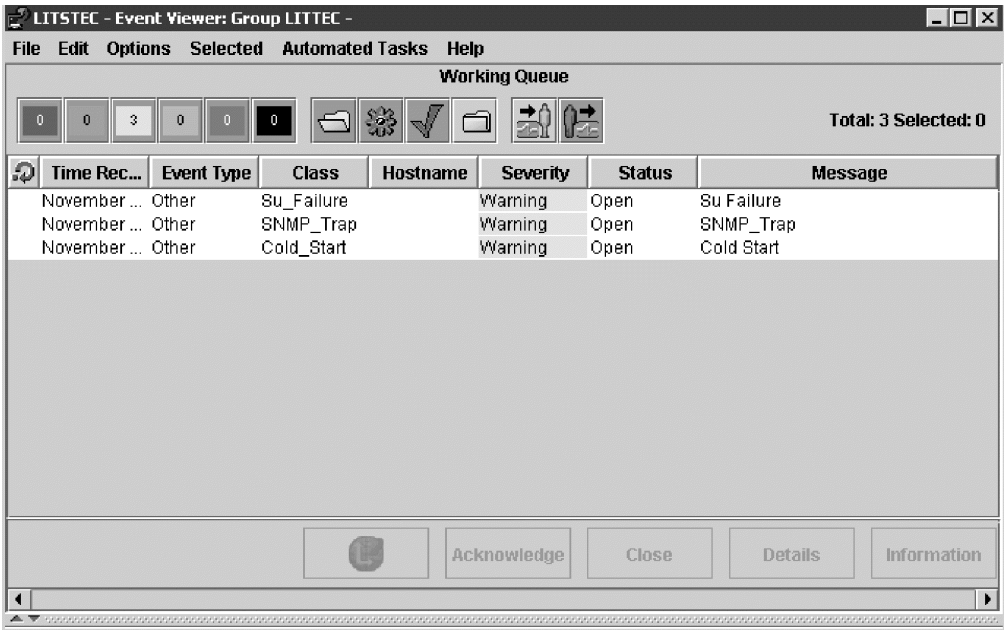


Figure 48. TEC Event Viewer display

Some useful commands

The following are some commands that you might find useful:

- Source the Tivoli environment before running any Tivoli commands: # . /etc/Tivoli/setup_env.sh
- To update the resource database in TMR after an uninstall of a product:

  ```
  # wchkdb -u
  ```
- To recycle the TMF server:

  ```
  # odadmin reexec all
  ```
- To list the servers in the TMR regions:

  ```
  # odadmin odlist
  ```
- To test the RIM object:

  ```
  # wrimtest -l tec
  ```
- To stop an event server:

  ```
  # wstopesvr
  ```
- To start an event server:

  ```
  # wstartesvr
  ```
- To see all the events in events database:

  ```
  # wtdumprl
  ```
- To clear all events in the events database:

  ```
  # wtdbclear -el -t 0
  ```
- To check space in the events database:

  ```
  # wtdbspace
  ```
- To list the endpoints nodes:

  ```
  # wep ls
  ```
- To find out the status of an endpoint node:

  ```
  # wepstatus litrwas4-ep
  ```

Tivoli Enterprise Console References

- *IBM Tivoli Enterprise Console Adapters Guide*, SC32-1242
- *IBM Tivoli Enterprise Console Command and Task Reference*, SC32-1232
- *IBM Tivoli Enterprise Console Installation Guide*, SC32-1233
- *IBM Tivoli Enterprise Console Release Notes*, SC32-1238
- *IBM Tivoli Enterprise Console Rule Developer's Guide*, SC32-1234
- *IBM Tivoli Enterprise Console Rule Set Reference*, SC32-1282
- *IBM Tivoli Enterprise Console User's Guide*, SC32-1235
- *Tivoli Event Integration Facility Reference*, SC32-1241

Chapter 14. Automation with Operations Manager for z/VM

As your environment evolves due to growth and customer demand, the resource requirements to manage the environment grow exponentially. Performing routine tasks required to keep your data center running smoothly can become an overwhelming and time consuming task. Often system administrators need to find ways of doing more with the existing resources they have, and many turn to automation of repetitive tasks to lessen their workload. Automation is one of most cost effective avenues one can take to increase productivity. Routine daily tasks such as health monitoring, log deletion, and backups are prime candidates for automation. We will share our experience implementing a systems automation tool, Operations Manager for z/VM, so that we, too, can continue to grow our environment.

Overview of Operations Manager

Operations Manager for z/VM is part of a set of systems management tools developed by IBM for z/VM. Since z/VM is a crucial part of our integrated lab testing environment, underlying many of our virtual machines, it was only natural for us to look at software to aid our day-to-day operations at this level in the software stack.

Operations Manager is the z/VM automation tool and monitoring tool. As such, you can view Operations Manager as a sophisticated console message viewing, and logging tool in one integrated package. Operations Manager can schedule tasks, perform rule-based message monitoring, and take action based on user-defined statements. Operations Manager has the ability to dynamically define, alter, or delete rules, actions, and scheduled events from a command line interface allowing for additional flexibility and can be used to monitor its own health. In this test report, we will share experiences and insights with setting up Operations Manager in our z/VM environment. In our laboratory environment we used Operations manager to automate routine tasks, setup console message views, and take action to recover ailing systems.

Before proceeding with the details of Operations Manager, it is worth noting that it has three sibling products named Archive Manager, Tape Manager, and Backup Manager. All three of these tools were designed from the ground up with their focus on the need for a cohesive set of systems management tools for z/VM. This set of tools can run independently or as an integrated set. We will only address Operations Manager in this report.

Planning to install Operations Manager

While reviewing the Program Directory, we noticed we needed some prerequisite software. Operations Manager requires the IBM Library for REXX on zSeries V1.4 (5695-014) to be installed. Though not covered in detail in this report, the installation of the requisite REXX Library was straight forward, though the system programmer responsible for this task should have a basic understanding of z/VM saved segments. The program directory for the IBM Library for REXX has clear, step-by-step instructions on how to do this. The instructions include using the VMFSGMAP tool to assist you in defining the segment. We found the VMFSGMAP tool very helpful for new z/VM system programmers when defining the segment.

Installing Operations Manager

The Program Directory for Operations Manager is well written and the installation process is straight forward. Any system programmer with a basic understanding of z/VM, the z/VM directory, and their locally deployed security manager can complete the installation with ease.

Configuring and testing Operations Manager

Operations Manager is made up of two types of service machines. The main service machine, OPMGRM1, runs the GOMMAIN module and is the central message processing service. Once started, GOMMAIN reads the Operations Manager configuration file containing authorization statements, schedules, rules, actions, and views. It then receives all monitored console messages (via SECUSER or OBSERVER), determines which rules to trigger based on the monitored messages, triggers schedules, and either performs actions locally or offloads them to an action processing server.

The other service machines, OPMGRM1-OPMGRM4, are the action processing servers. They run the GOMSVM module and handle offloaded processing of actions. We will discuss offloading processes in more detail when covering automating processes.

Operations Manager for z/VM Administration Guide contains the reference information needed to configure Operations Manager. However, like any other product, until you have gone through an installation yourself, you may not quite grasp how all the pieces fit together. In "Starting Operations Manager for the first time," we will discuss the steps performed to deploy Operations Manager into our simulated production environment.

Starting Operations Manager for the first time

When we initially tried to load the sample configuration file, OPMGRM1 CONFSAMP, on OMPGRM1, we were unable to locate it on our system. As it turns out, the sample file disk was not linked by default. To remedy this, we simply added a link to the sample disk, 5697J10B 2C2, in OPMGRM1's directory entry and accessed the disk in the PROFILE EXEC.

For a quick test, we chose to use DEFMMON to monitor a z/VM guest. The DEFMMON is used to define a monitor in the configuration file. Its primary function is to see if a virtual machine is running and if not, trigger a user defined action. In addition, DEFMMON can check on dependencies before triggering an action, define a delay time to wait before starting the action, define what days and times the action should be in effect, and define the interval on which the monitor checks the status of the virtual machine that it is monitoring.

In our test, we defined a DEFMMON statement in the sample configuration file which would monitor a VSWITCH controller, DTCVSW1 and XAUTOLOG it if not available. In addition to the action, we specified a dependency on the TCPIP service machine. If the dependency is not met, actions are not triggered. Below is the DEFMMON statement we used.

```
DEFMMON NAME(DTCVSW1),USER(DTCVSW1),ACTION(XAUTOLOG),+
         PARM(DTCVSW1),DEPEND(TCPIP,A)
```

Note that the + at the end of the line specifies that the statement continues on the next line.

With this configuration in place, Operations Manager is ready to be started and subsequently tested. We started Operations Manager from the OPMGRM1 user ID using the following command:

```
GOMMAIN OPMGRM1 CONFSAMP
```

Once initialized, we forced the DTCVSW1 machine offline from the MAINT user ID. Operations Manager detected DTCVSW1 was no longer running and triggered the XAUTOLOG action as expected. The following is the log of the event:

```
GOMACT0260I MONITOR DTCVSW1 ACTION XAUTOLOG TRIGGERED BY MONITOR
GOMACT0262I ACTION XAUTOLOG BEGIN FOR MONITOR SERVER OPMGRM1
GOMACT0269I COMMAND "CP XAUTOLOG DTCVSW1"
GOMACT0267I ACTION XAUTOLOG END RC=0 SERVER OPMGRM1
GOMACT0263L Command accepted
GOMCMD0216L "RACFVM ICH70001I DTCVSW1 LAST ACCESS AT 09:09:48 ON THU
GOMCMD0216L "OPMGRM1 AUTO LOGON *** DTCVSW1 USERS = 52" VID=*M
GOMCMD0216L "OPMGRM1 HCPCLS6056I XAUTOLOG information for DTCVSW1:
```

As you can clearly infer from the above log, Operation Manager was installed and configured correctly. With the basic deployment operational, site-specific alterations for the environment can now be performed.

Customizing Operations Manager for our environment

We keep a number of configuration files on the USER 191 disk. It is linked RR by our service machines and MR by MAINT for updates and service. Ownership of the disk belongs to the NOLOG user ID, USER. This is the same ID that contains a number of disks with local utilities and files we share on our system. You may recall that we used this ID in our June 2005 test report in the section on Cloning Linux Images on z/VM. There we used the USER 194 and 195 disks to hold our programs and configuration files needed for the cloning process.

To adapt Operations Manager to our environment, we added a read link to the USER 191 disk as OPMGRM1 291 in the directory. We access it as the B-disk in the PROFILE EXEC for OPMGRM1. Sharing a single disk for a number of different configuration files makes system management much easier by placing everything in one location. This way, when we need to update a file for any of our service machines, we only need to link to a single disk from MAINT. This also allows for making changes without having to log onto the service machine, reducing down time.

In addition to setting up the shared configuration disk, we needed to define OPMGRM1 as the secondary console for the systems that will have their console messages monitored by Operations Manager. Our goal is to monitor all Linux guests, so we have added a secondary console statement in the directory profile, LINDFLT, which is used by our Linux systems.

The following is an example of our LINDFLT directory profile:

```
PROFILE LINDFLT
 CRYPTO APVIRT
 IUCV ALLOW
 SPOOL 00C 2540 READER
 SPOOL 00D 2540 PUNCH
 SPOOL 00E 1403 A
 CONSOLE 009 3215 T OPMGRM1
 LINK MAINT 0190 0190 RR
 LINK MAINT 019D 019D RR
 LINK MAINT 019E 019E RR
 LINK USER  0196 0196 RR
```

Using Operations Manager

We'll discuss several aspects of how we put Operations Manager to work in our environment.

Using XAUTOLOG with Operations Manager

You might recall from our December 2006 Test Report that we used AUTOLOG2 to start our Linux guests in a particular order after the z/VM system is IPLed. To take advantage of the flexibility and monitoring features Operations Manager provides, we have implemented the XAUTOLOG process with the DEFMMON statement we showed you in "Starting Operations Manager for the first time" on page 166. Utilizing the action, delay, and dependency features of DEFMMON, we can specify the order in which the systems are IPLed, make sure that any dependent systems are running first, and take action if the system was to go down with this single command.

The DELAY operand of DEFMMON specifies how long Operations Manager is to execute before checking the status of the monitored virtual machine. For our deployment, Operations Manager is configured to start execution immediately after the system is IPLed (This is done via the typical AUTOLOG1 mechanism).

From there, once Operations Manager is initialized, DEFMMON statements are processed. As each statement is executed, the monitor detects which virtual machines are not logged on and triggers their corresponding defined XAUTOLOG action once the specified delay time has elapsed. Using DEFMMON in this fashion gives us tight control over the order of the IPL procession.

If additional systems are added to your environment or if you need to change the sequence in which they are started, only the value of the DELAY operand needs to be altered. The position of the DEFMMON statements in the configurations is secondary to the value of the DELAY statement.

The DEPEND operand is also quite important. The DEPEND operand instructs DEFMMON to check the status of the specified virtual machine before performing the action listed in the ACTION operand. In our deployment, the VSWITCH controller is of no use if the TCPIP stack is not operational. By specifying TCPIP on the DEPEND operand, the VSWITCH controller will not be XAUTOLOGed unless TCPIP is running.

In the following example, the sequence in which the systems are autologged is primarily determined by the value of the DELAY operand. This is why LITHUB, LITSGFW1, and LITSGFW2 start before LITSTAT2, LITSTAM2, and LITDCON1. Also note in the following example that the VSWITCH controllers, DTCVSW1 and DTCVSW2, will not start until after TCPIP has been autologged due to the additional DEPEND operand.

```
************************************************************
* This is the XAUTOLOG and MONITOR section                 *
************************************************************
DEFMMON NAME(OPMGRS1),USER(OPMGRS1),ACTION(XAUTOLOG),PARM(OPMGRS1)
DEFMMON NAME(PERFSVM),USER(PERFSVM),ACTION(XAUTOLOG),PARM(PERFSVM)
DEFMMON NAME(DTCVSW1),USER(DTCVSW1),ACTION(XAUTOLOG),PARM(DTCVSW1),+
 DEPEND(TCPIP,A),DELAY(1)
DEFMMON NAME(DTCVSW2),USER(DTCVSW2),ACTION(XAUTOLOG),PARM(DTCVSW2),+
  DEPEND(TCPIP,A),DELAY(1)
DEFMMON NAME(LITSTAT2),USER(LITSTAT2),ACTION(XAUTOLOG),+
  PARM(LITSTAT2),DELAY(3)
```

```
DEFMMON NAME(LITSTAM2),USER(LITSTAM2),ACTION(XAUTOLOG),+
  PARM(LITSTAM2),DELAY(3)
DEFMMON NAME(LITDCON1),USER(LITDCON1),ACTION(XAUTOLOG),+
  PARM(LITDCON1),DELAY(3)
DEFMMON NAME(LITHUB),USER(LITHUB),ACTION(XAUTOLOG),PARM(LITHUB),+
  DELAY(2)
DEFMMON NAME(FW1),USER(LITSGFW1),ACTION(XAUTOLOG),PARM(LITSGFW1),+
  DELAY(2)
DEFMMON NAME(FW2),USER(LITSGFW2),ACTION(XAUTOLOG),PARM(LITSGFW2),+
  DELAY(2)
```

At this time, some readers might be wondering why we did not employ heavier use of the DEPEND operand on more of our Linux guests. Quite simply, we need not use it as often because of the HA virtualized environment where the Linux servers are deployed. Due to the HA configuration spread across different z/VM systems and LPARs, hard dependency among Linux servers is not common.

Scheduling tasks in Operations Manager

We have set up Operations Manager to run a number of health checking routines. Many of these checks are relatively simple but their output can be invaluable when debugging a problem. For instance, checking paths to DASD, querying spool and page space, and running the INDICATE command can be of great value to the system programmer when debugging a problem. It can be especially useful when a system crashes and an overzealous operator restarts it without collecting detailed documentation. We use a combination of DEFSCHD and DEFACTN to accomplish this.

The DEFSCHD statement is used to define a scheduled event. Events can be scheduled to run at a specific time or on an interval. The scheduler is aware of the day of week and holidays.

The DEFACTN statement is used to define an action and specifies what commands or programs are to be executed.

One example is how we check paths to our critical DASD. This requires checking paths on DASD 3000, 3100, 3200, and 3300 devices. Since each volume on the same controller uses the same set of paths, we only need to check the first volume. In addition, since the DEFACTN function provides a NEXTACTN operand, it was only necessary to create a single scheduled event and then have each action call the next using NEXTACTN. The NEXTACTN operand specifies the next action to execute when the current action completes. You can chain a number of actions together with the NEXTACTN. However, you cannot define an action as NEXTACTN that has already been used in the chain.

In some situations, for the purpose of readability, statements are grouped together in the configuration file by task. Operations Manager provides this flexibility by allowing you to mix command order in the configuration file. It should be noted that doing so can have performance implications which we will discuss in more detail in “Operations Manager performance” on page 174. In this case, keeping the file readable is more important then performance.

The following are sample DEFSCHD and DEFACTN statements to check paths to our DASD:

```
*
*  Check PATHS to dasd after IPL and ever 6 hours.
*
```

```
DEFSCHD NAME(CKPATH),EVERY(06:00),ACTION(CK3000)
DEFACTN NAME(CK3000),COMMAND(CP Q PATH 3000),NEXTACTN(CK3100),+
  INPUT(LOG,ARV),OUTPUT(LOG,ARV),ENV(LVM)
DEFACTN NAME(CK3100),COMMAND(CP Q PATH 3100),NEXTACTN(CK3200),+
  INPUT(LOG,ARV),OUTPUT(LOG,ARV),ENV(LVM)
DEFACTN NAME(CK3200),COMMAND(CP Q PATH 3200),NEXTACTN(CK3300),+
  INPUT(LOG,ARV),OUTPUT(LOG,ARV),ENV(LVM)
DEFACTN NAME(CK3300),COMMAND(CP Q PATH 3300),+
  INPUT(LOG,ARV),OUTPUT(LOG,ARV),ENV(LVM)
```

In addition to simply using the CONFIG statement within the main configuration file, Operations Manager also allows you to embed other configuration files. This functionality allows you to group and manage related tasks in smaller, more focused configuration files which are in turn included by the main configuration file. For example, we could have placed the statement:

```
CONFIG FN(CHECKDSK),FT(CONFIG)
```

in the main configuration and created a file named CHECKDSK CONFIG with the statements above with equivalent results.

Several tools used for monitoring and triggering actions can only process one task at a time. Executing long running tasks presents a problem because while the tool is executing your task, it is not processing incoming messages that may be critical. Operations Manager has eliminated this problem by allowing the triggered actions to execute remotely on one of the action processing servers (OPMGRS1-OPMGRS4).

To enable remote execution, simply specify the ENV(SVM) operand on the action statement. Running the program remotely also serves to prevent buggy software or test programs from hanging your automation tool.

An example use of the remote execution is performed in our environment to perform nightly backups of the USER 191 disk. This scheduled event, called BACKDSKS, is set to run at 1:00 AM. The action BACKDSKA runs the command EXEC, which in turn calls the REXX program BACK191. Since the operand ENV(SVM) is specified as an action attribute, the REXX program executes remotely on one of the OPMGRS*x* service machines. We will show this example in a moment.

In addition to specifying the location of execution, we can optionally display the output from the backup job in a console view we define in addition to placing the output into the Operations Manager log. We will discuss defining views in "Defining views in Operations Manager" on page 172. We define the output attributes by using the OUTPUT(CYE,LOG) operand on the action statement, which completes the following example:

```
DEFSCHD NAME(BACKDSKS),WHEN(01:00),ACTION(BACK191)
DEFACTN NAME(BACKDSKA),COMMAND(EXEC BACK191),+
  OUTPUT(CYE,LOG),ENV(SVM)
```

Notes:

1. For readability within this report, we named the scheduled event BACKDSKS and the action BACKDSKA. In the original configuration file that we used internally, both were named BACKDSK. It is valid to name your Operations Manager actions, schedules and other types with identical names, though clarity may suffer.
2. Names used in Operations Manager are limited to 8 characters. With a little creativity in naming, this should not be an issue.

For fine-grained control with the ENV operand, Operations Manager supports the ENV(*userid*) variant, which allows you to assign to an action a specific server for processing. This may be exceptionally useful when the requested server has specific requisite privileges or some other special means of data access required for execution.

Defining rules in Operations Manager

Rules bring out the real power of Operations Manager. They allow you to automate responses, highlight messages, take preventative action, or recover from something bad that has happened.

For example, here in the laboratory, we wanted certain high-priority messages from our Linux systems to be highlighted on the console view in order to stand out. To do this, we first created an action called ALERT which displays the message on the console in yellow. Next, we created a number of rules to monitor the messages and trigger the ALERT action when a match occurs.

For instance, in our environment, the crypto facility is important to our HTTP servers using SSL. To prevent a server from running unnoticed without the crypto facility, we created a rule to check for message:

```
HCPKFL663E, system unable to find the crypto facility.
```

When such a message is found, the rule triggers the ALERT action. Rules are defined with the DEFRULE command. Here is an example of the rule we defined to check for the crypto message:

```
DEFRULE NAME(CRYPFAIL),MATCH(*HCPKFL663E*),ACTION(ALERT)
```

Simply highlighting a message on the console may not be enough for your deployment. The action ALERT can also be modified in other ways, such as to send an email to the system programmer with a notification of the problem.

The sample configuration file that ships with Operations Manager features a number of useful rules that you may want to investigate. One of particular interest to us is PSWWAIT. This rule detects when a Linux system has entered a wait state and will subsequently re-IPL the system. This rule integrates well with our existing HA recovery schema. Below is the sample rule and action for the PSWWAIT as deployed here:

```
DEFRULE NAME(PSWWAIT),MATCH(*HCP*450W*),ACTION(PSWWAIT)
DEFACTN NAME(PSWWAIT),COMMAND(CP SEND &U IPL CMS),+
   INPUT(LOG,CBL),OUTPUT(LOG),ENV(LVM)
```

Viewing logs in Operations Manager

Logging is another crucial component of Operations Manager functionality. Operations Manager provides functionality to have virtual machines view the Operations Manager aggregate log. This log contains the console output of all monitored consoles in addition to any Operations Manager messages indicating when rules or schedules have been triggered, actions are executed, and similar interesting operations manager events occur. To view this log you use the VIEWLOG function.

Virtual machines authorized to use VIEWLOG can perform two functions. If issued without any parameters, the current copy of the log will be displayed on your terminal. However, VIEWLOG does not display messages with attributes set via the OUTPUT operand. The second option is to recall historical logs by supplying the log date as a parameter. VIEWLOG will send the log to your RDR. If you

specify the CLASS parameter, VIEWLOG will use that class when sending the log to you. Access to the GOMCMD is needed to issue the VIEWLOG function.

The syntax of VIEWLOG to view the log in real time from a virtual machine running CMS is:

```
GOMCMD OPMGRM1 VIEWLOG
```

To reduce the amount of typing needed to issue VIEWLOG, especially when recalling logs, we created the following utility REXX executable:

```
/* VIEWLOG EXEC */
/* Call Operations Manager VIEWLOG    */
/* Useage:  VIEWLOG  [date] [class]  */
arg logdate logclass
if logdate ="" then do
   'GOMCMD OPMGRM1 VIEWLOG'
   EXIT
END
if logclass ='' then 'GOMCMD OPMGRM1 VIEWLOG DATE('logdate')'
ELSE 'GOMCMD OPMGRM1 VIEWLOG DATE('logdate'),CLASS('logclass')'
```

Defining views in Operations Manager

Viewing live console messages, though crucial to any complex environment, may be daunting. When you consolidate the view of many consoles without a proper filtering mechanism, verbose messages can prove to be unwieldy. Operations Manager includes functionality allowing you to group live consoles messages and interact with them from a single, well organized consolidated view.

In addition to VIEWLOG, which allows you to view all messages processed by Operations Manager, the VIEWCON command allows you to define which virtual machine console you want to monitor and interact with (in approximate real time). VIEWCON also applies display attributes like color and hold set to messages via the DEFACTN operands, INPUT and OUTPUT. This is useful when you need to alert the operator or system programmer to something happening on your system.

For instance, in our example of when disk backup jobs were executed (in "Scheduling tasks in Operations Manager" on page 169), the output from the job was passed on to the console in yellow. Another example use is using red color on the console to denote when a Linux system enters CP READ. Message attributes are controlled by the INPUT and OUTPUT operands on the DEFACTN command.

In order to view a console, a user needs to be authorized. This is done with the AUTH USER command. In the following example, we have set up two console views: One to monitor all of our systems, named ALL, and another to view only messages from our HTTP servers, named WAS:

```
* Define CONSOLE VIEW
DEFVIEW NAME(ALL),USER(*)
DEFVIEW NAME(WAS),USER(*WAS*)
*
* Define users with control authority to Operations Manager
*
AUTH USER(BEYER),CONSOLE(ALL)
AUTH USER(BEYER),CONSOLE(WAS)
AUTH USER(BEYER),CONTROL(Y)
AUTH USER(OPMGRM1),CONTROL(Y)
AUTH USER(MAINT),CONTROL(Y)
*
```

You may wonder why we have defined a console view that can see all messages when VIEWLOG does just this. As we stated earlier, VIEWLOG does not use the display attributes when displaying the message; only VIEWCON will do that.

Using VIEWCON

Just like VIEWLOG, VIEWCON is issued via the GOMCMD command. The syntax for the VIEWCON in the Administration Guide is a bit misleading. It states the operand for USER is the user ID you want to view. While this is true, you can instead chose to specify the name of a view you defined with DEFVIEW.

For example, to display the defined view named WAS, issue:

```
GOMCMD OPMGRM1 VIEWCON  USER(WAS)
```

By default, when the view appears on your 3270 terminal, it does not display the date and time. Simply pressing F6 will toggle between, no date or time, time, and date and time, as shown in Figure 49.

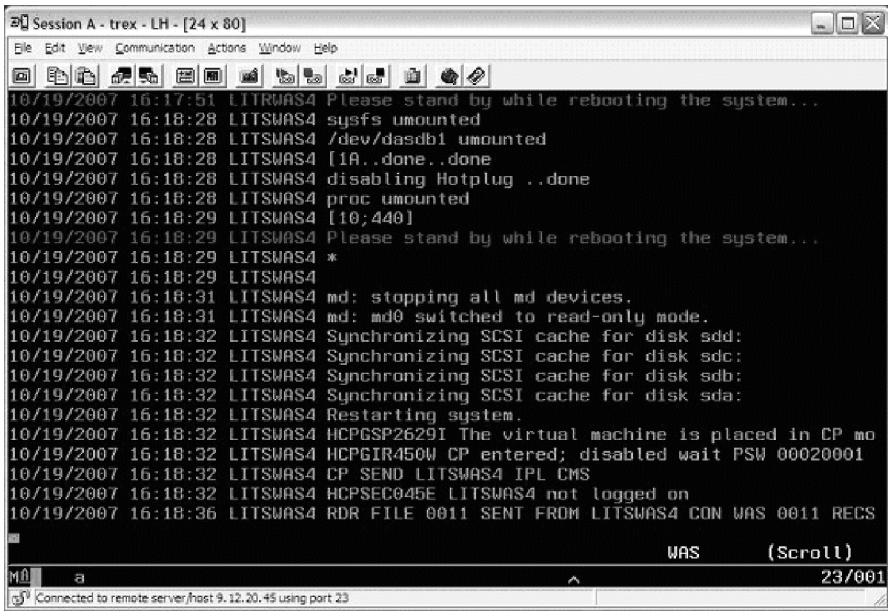


Figure 49. VIEWCON view with both date and time enabled

Note that you can set up a VIEWCON PROFILE that will set your default date and time format preference. You can also display what each PF key does from a profile.

Just like we did for the VIEWLOG command, we created a REXX EXEC to start VIEWCON with the WAS view:

```
/* VIEWWAS  EXEC */
/* Operations Manager VIEWCON – WAS view*/
'GOMCMD OPMGRM1 VIEWCON  USER(WAS)'
```

Dynamic Configuration

If you are already using a different systems management and automation tool, you may ask why you should implement Operations Manager. Simply put, the answer is: dynamic configuration. Operations Manager has the ability to dynamically update its configuration. This means that you can define, delete, and update rules, actions, and schedules on the fly via the command line from a console or in an EXEC (with no restarts required).

This functionality is especially handy when adding new systems to your monitoring environment. Traditionally, you would update the configuration file then schedule a time when you could restart the monitor reading in the new configuration. Meanwhile, if the new system encountered a problem, it would potentially go unnoticed by the monitor. With Operations Manager, monitoring can be enabled instantly via the DEFMMON command followed by the traditional method of adding the statements to the configuration file. This capability eliminates the exposure between the time the system was added and the monitor was updated.

The real power of dynamic configuration is realized when you write automated REXX EXECs to perform tasks. Refer to our backup example in "Scheduling tasks in Operations Manager" on page 169. Before you can start the backup, you must typically shut down the guest to flush its I/O cache and ensure a consistent backup. With monitoring running, when the guest logs off the monitor will detect it and autolog the system. Left unchecked, this could prove problematic as it would restart the guest before the backup was able to complete. Therefore, as part of the backup job, the monitor needs to be temporarily disabled. This is accomplished by deleting the DEFMMON statement for the guest. An additional rule may be added at this time to detect when the backup has completed and reinstate the previously operational monitor. If one is sophisticated, they might also follow up with the removal of the rule to detect the backup job completion. Not only does this functionality make Operations Manager flexible, it also improves performance by not cluttering the configuration file with rules that are seldom used.

High availability and recoverability

High availability (HA) and recoverability are of great importance in the laboratory and in the field. HA is so important that it was the primary focus of our June 2007 test report. Operations Manager not only fits well in HA environments by defining rules and actions that automate system recovery, but it is also easy to make Operations Manager itself highly available.

In order to build an HA implementation of Operations Manager, simply set up a second Operations Manager guest which has only one function, the monitoring of the primary Operations Manager system, OPMGRM1. This can be done with a single DEFMMON monitor statement and an action statement to XAUTOLOG OPMGRM1 if it is not running. Should the primary instance fail, it will be restarted via this secondary watchdog implementation.

Operations Manager performance

Operations Manager processes the configuration file sequentially, from the top down. Though you may not have considered it, a vast amount of messages may be funneled through this rule base, each one tested sequentially. Therefore, you should strive to place the most commonly used rules at the top of the file to obtain

better performance. In addition, you can specify the FINAL(Y) operand on the DEFRULE statement. The FINAL operand specifies that message filtering should stop processing a message instance after the current rule has finished.

Gotchas

While implementing Operations Manager, we ran into one incident that puzzled us for a while. If you mistakenly place a space, instead of a comma, after the NAME in the DEFACTN statement, no error will be reported. However, the rest of the statement will be treated as a comment, never executing the intended action.

Overall, we found Operations Manager very easy to implement for our purposes. The free form structure of the configuration file aided readability by grouping rules, actions, and schedules together by function, while placing the most commonly used rules near the top of the file assisted in performance. The ability to offload processes to one of the OPMGRS*x* machines is also an invaluable benefit enabling system programmers to test new programs and REXX EXECs without concern for halting the entire automation tool.

Be sure to check future test reports to see how we use Operations Manager in some of the other phases of our ongoing systems management investigations.

Summary of our testing

During the phase of testing that has culminated in this test report, we implemented various tools focused on availability management. We have certainly achieved productivity gains as a result here in our laboratory environment and we hope many of the methodologies we have demonstrated can be applied in your enterprise, as well. The logging infrastructure recommendations provide centralized and consolidated log management, working in conjunction with dynamic monitoring which provide a central console. These new centralized information sources provide the input to advanced systems automation tools. Though we have only begun examining systems management tools, it is critical to have these less glamorous components in place. Far too often, these tasks languish behind those deemed higher in priority, yet they form the foundation of future automation endeavors that are so critical to the modern enterprise.

Chapter 15. Future Linux on System z projects

For the next phase of our testing, we plan to continue implementing system management architectures throughout our environment, enhancing the availability management processes put in place during this phase. We also intend to expand into other areas of systems management ,such as data management, security management, performance management, capacity management, and the related system programmer tasks.

We also plan to continue our tradition of documenting the upgrades to our production Linux distributions, along with the middleware deployed on them, to the current supported levels. During this upgrade cycle, we hope to document a methodology which uses revolving upgrades in place, as we are now positioned with the HA and RAS-BR implementations covered in the previous two test reports.

As always, feel free to send us your comments, questions, or areas of interest!

Appendix A. Some of our RMF reports

We provide the following examples of some of our RMF reports:

RMF Monitor I Post Processor Summary

The following contains information from our RMF *Monitor I Post Processor Summary Report*. Some of the information we focus on in this report includes CP (CPU) busy percentages and I/O (DASD) rates.

```
                                    R M F   S U M M A R Y   R E P O R T
                                                                                    PAGE  001
             z/OS  V1R9               SYSTEM ID J80             START 07/17/2007-10.45.00  INTERVAL 00.14.59
                                      RPT VERSION V1R9 RMF      END   07/17/2007-11.00.00  CYCLE 0.100 SECONDS

 NUMBER OF INTERVALS 1
 DATE    TIME     INT    CPU   DASD  DASD  TAPE   JOB   JOB   TSO   TSO   STC   STC  ASCH  ASCH  OMVS  OMVS SWAP DEMAND
 MM/DD HH.MM.SS MM.SS  BUSY  RESP  RATE  RATE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE RATE PAGING
 07/17 10.45.00 14.59  89.9   2.6 18474   0.0   176   175    42    41   493   485     2     0    49    38 0.00   0.02


                                    R M F   S U M M A R Y   R E P O R T
                                                                                    PAGE  001
             z/OS  V1R9               SYSTEM ID J90             START 07/17/2007-10.45.00  INTERVAL 00.15.00
                                      RPT VERSION V1R9 RMF      END   07/17/2007-11.00.00  CYCLE 0.100 SECONDS

 NUMBER OF INTERVALS 1
 DATE    TIME     INT    CPU   DASD  DASD  TAPE   JOB   JOB   TSO   TSO   STC   STC  ASCH  ASCH  OMVS  OMVS SWAP DEMAND
 MM/DD HH.MM.SS MM.SS  BUSY  RESP  RATE  RATE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE RATE PAGING
 07/17 10.45.00 15.00  72.1   1.8 12678   0.0   177   174    14    14   386   381     1     0    23    18 0.00   0.00


                                    R M F   S U M M A R Y   R E P O R T
                                                                                    PAGE  001
             z/OS  V1R9               SYSTEM ID JB0             START 07/17/2007-10.45.00  INTERVAL 00.15.00
                                      RPT VERSION V1R9 RMF      END   07/17/2007-11.00.00  CYCLE 0.100 SECONDS

 NUMBER OF INTERVALS 1
 DATE    TIME     INT    CPU   DASD  DASD  TAPE   JOB   JOB   TSO   TSO   STC   STC  ASCH  ASCH  OMVS  OMVS SWAP DEMAND
 MM/DD HH.MM.SS MM.SS  BUSY  RESP  RATE  RATE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE RATE PAGING
 07/17 10.45.00 15.00  76.5   1.9 10608  1213   179   178     7     7   690   685     0     0    29    22 0.00   0.00


                                    R M F   S U M M A R Y   R E P O R T
                                                                                    PAGE  001
             z/OS  V1R9               SYSTEM ID JC0             START 07/17/2007-10.45.00  INTERVAL 00.15.00
                                      RPT VERSION V1R9 RMF      END   07/17/2007-11.00.00  CYCLE 0.100 SECONDS

 NUMBER OF INTERVALS 1
 DATE    TIME     INT    CPU   DASD  DASD  TAPE   JOB   JOB   TSO   TSO   STC   STC  ASCH  ASCH  OMVS  OMVS SWAP DEMAND
 MM/DD HH.MM.SS MM.SS  BUSY  RESP  RATE  RATE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE RATE PAGING
 07/17 10.45.00 15.00  65.5   6.4 994.1   0.0   175   174     0     0   416   413     1     0    24    19 0.00   0.02


                                    R M F   S U M M A R Y   R E P O R T
                                                                                    PAGE  001
             z/OS  V1R9               SYSTEM ID JA0             START 07/17/2007-10.45.00  INTERVAL 00.14.59
                                      RPT VERSION V1R9 RMF      END   07/17/2007-11.00.00  CYCLE 0.100 SECONDS

 NUMBER OF INTERVALS 1
 DATE    TIME     INT    CPU   DASD  DASD  TAPE   JOB   JOB   TSO   TSO   STC   STC  ASCH  ASCH  OMVS  OMVS SWAP DEMAND
 MM/DD HH.MM.SS MM.SS  BUSY  RESP  RATE  RATE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE RATE PAGING
 07/17 10.45.00 14.59  81.3   3.0 12716   0.0    33    31    32    32   420   407     2     0    85    70 0.00   0.00
```

```
                                    R M F   S U M M A R Y   R E P O R T
                                                                                PAGE  001
            z/OS  V1R9              SYSTEM ID JE0             START 07/17/2007-10.45.00  INTERVAL 00.15.00
                                    RPT VERSION V1R9 RMF      END   07/17/2007-11.00.00  CYCLE 0.100 SECONDS

 NUMBER OF INTERVALS 1
 DATE   TIME     INT   CPU   DASD  DASD  TAPE   JOB   JOB   TSO   TSO   STC   STC  ASCH  ASCH  OMVS  OMVS SWAP DEMAND
 MM/DD HH.MM.SS MM.SS  BUSY  RESP  RATE  RATE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE RATE PAGING
 07/17 10.45.00 15.00  26.1   1.0  4660   0.0     3     1     1     1   371   366     1     0    39    28 0.00   0.00


                                    R M F   S U M M A R Y   R E P O R T
                                                                                PAGE  001
            z/OS  V1R9              SYSTEM ID JF0             START 07/17/2007-10.45.00  INTERVAL 00.15.00
                                    RPT VERSION V1R9 RMF      END   07/17/2007-11.00.00  CYCLE 0.100 SECONDS

 NUMBER OF INTERVALS 1
 DATE   TIME     INT   CPU   DASD  DASD  TAPE   JOB   JOB   TSO   TSO   STC   STC  ASCH  ASCH  OMVS  OMVS SWAP DEMAND
 MM/DD HH.MM.SS MM.SS  BUSY  RESP  RATE  RATE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE   MAX   AVE RATE PAGING
 07/17 10.45.00 15.00   8.3   4.4 293.5   0.0     3     1     1     1   389   387     1     0    36    28 0.00   0.00
```

RMF Monitor III Online Sysplex Summary

The following contains information from the RMF *Monitor III Online Sysplex Summary*. This is a real-time report available if you are running WLM in goal mode. We highlighted some of our goals and actuals for various service classes and workloads. At the time this report was captured we were running 1972 CICS transactions/second.

```
             HARDCOPY  RMF V1R9   Sysplex Summary - UTCPLXJ8       Line 1 of 84
 Command ===>                                                    Scroll ===> CSR
 WLM Samples: 239     Systems: 7  Date: 07/17/07 Time: 10.45.00 Range: 60    Sec

                      >>>>>>>>XXXXXXXXXXXXXXXXXX<<<<<<<<

 Service Definition: WLMDEF02              Installed at: 07/12/07, 20.00.32
      Active Policy: WLMPOL01              Activated at: 07/12/07, 20.00.40
                 ------- Goals versus Actuals --------  Trans --Avg. Resp. Time-
                 Exec Vel  --- Response Time ---  Perf  Ended  WAIT EXECUT ACTUAL
 Name     T  I   Goal Act  ---Goal--- --Actual--  Indx  Rate   Time   Time   Time
 BATCH    W           30                            0.117 1.662  2.12M  2.13M
 BATI1V90 S  1    90  69                      1.31  0.000 0.000  0.000  0.000
 BATI2V50 S  2    50  66                      0.76  0.000
 DISCR    S  D       9.6                           0.117
 CICS     W           71                           1972 0.000  0.843  0.982
 CI2V60   S  2    60  71                      0.85  0.033
 CI280%P6 S  2       N/A  0.600 80%         95%  0.50   1471 0.000  0.201  0.426
 CI350%10 S  3       N/A  10.00 50%         95%  0.70  2.150 0.000  6.081  6.081
 CI390%01 S  3       N/A  1.000 90%         85%  **** 498.6 0.000  2.282  1.137
 ICSS     W           42                           51.25 0.001  0.086  0.086
 ICI2V50  S  2    50  70                      0.71  41.02 0.001  0.092  0.093
 IC14V50  S  4    50  13                      3.75  10.23 0.001  0.060  0.060
 IMS      W          N/A                           98.10 0.000  0.349  0.606
 II290%P5 S  2       N/A  0.500 90%         94%  0.70  31.38 0.000  0.158  0.177
 II390%P7 S  3       N/A  0.700 90%         69%  2.00  65.28 0.000  0.444  0.824
 II490%01 S  4       N/A  1.000 90%        100%  0.50  1.433 0.000  0.108  0.114
 STC      W          34                           27.70 0.001  3.997  3.998
 STCI1V40 S  1    40  90                      0.44  0.400 0.002  0.256  0.258
 STCI1V90 S  1    90  60                      1.50  0.000 0.000  0.000  0.000
 STCI2V30 S  2    30  10                      2.90  0.000
 STCI2V40 S  2    40  68                      0.59  0.233 0.002  0.509  0.511
 STCI2V50 S  2    50  74                      0.67  0.000
```

```
|  STCI2V60 S  2    60  71                          0.85  0.000
|  STCI2V70 S  2    70  85                          0.82  0.000 0.000  0.000  0.000
|  STCI5V05 S  5     5  61                          0.08  0.750 0.000  0.019  0.019
|  STCI5V10 S           16                                0.000 0.000  0.000  0.000
|           3  5    10  16                          0.63  0.000 0.000  0.000  0.000
|  STCOMVS  S           17                                26.32 0.001  4.198  4.199
|           1  2    30  44                          0.68  4.900 0.000  2.468  2.468
|           2  3    20  58                          0.34  7.600 0.000  3.730  3.730
|           3  5    10  16                          0.64  13.82 0.002  5.070  5.072
|  SYSTEM   W           78                                0.017 0.432  0.663  1.095
|  SYSSTC   S      N/A  83    N/A                         0.017
|  SYSTEM   S      N/A  68    N/A                         0.000 0.000  0.000  0.000
|  TSO      W           51                                8.283 0.000  6.840  6.840
|  TSO      S  2        51  2.000 AVG  6.840  AVG  3.42  8.283 0.000  6.840  6.840
|  WAS      W           11                                278.0 0.181  0.478  0.659
|  WI1VEL30 S  1    30  81                          0.37  0.000 0.000  0.000  0.000
|  WI180%01 S  1       9.5  1.000 80%          81%  1.00  278.0 0.181  0.478  0.659
|  WI2VEL50 S  2    50  22                          2.33  0.000
|  CICSCONV R          N/A                               2.150 0.000  6.081  6.081
|  CICSCPSM R          N/A                               1.200 0.000  0.002  0.002
|  CICSMISC R          N/A                               213.0 0.000  0.036  0.036
|  CICSWEB  R           46                               0.000 0.000  0.000  0.000
|  Name     T  I  Goal Act  ---Goal--- --Actual--  Indx  Rate   Time   Time   Time
|  CQSREP   R           71                                0.000 0.000  0.000  0.000
|  CSQCCHIN R           76                                0.000 0.000  0.000  0.000
|  CSQCMSTR R           71                                0.000 0.000  0.000  0.000
|  DB2IRLM  R           90                                0.000 0.000  0.000  0.000
|  DB2REP   R           80                                0.000 0.000  0.000  0.000
|  DB2WLM   R           50                                0.000 0.000  0.000  0.000
|  DDF      R           61                                0.750 0.000  0.019  0.019
|  EWLMWORK R           48                                49.72 0.002  0.156  0.158
|  FDBRREP  R           40                                0.000 0.000  0.000  0.000
|  GRS      R           76                                0.000 0.000  0.000  0.000
|  IMSREG   R           60                                0.000 0.000  0.000  0.000
|  IMSREP   R           73                                0.000 0.000  0.000  0.000
|  MQ       R           43                                0.000 0.000  0.000  0.000
|  OMVS     R           77                                0.000 0.000  0.000  0.000
|  OMVSINIT R           90                                0.400 0.002  0.256  0.258
|  OMVSTPNS R          0.0                                0.000 0.000  0.000  0.000
|           3       10 0.0                          N/A   0.000 0.000  0.000  0.000
|  RMF      R           97                                0.000 0.000  0.000  0.000
|  RMFGAT   R           83                                0.000 0.000  0.000  0.000
|  RWASCR   R           22                                0.000 0.000  0.000  0.000
|  RWASDFLT R          0.0                                82.78 0.024  0.145  0.170
|  RWASMED  R          0.0                                145.5 0.331  0.777  1.108
|  RWASWEB  R           81                                0.000 0.000  0.000  0.000
|  SLOWICSS R          0.0                                10.23 0.001  0.060  0.060
|  SYSSTC   R           49                                0.083 0.447  2.44H  2.44H
|           1           49                                0.083 0.447  2.44H  2.44H
|           3       10  16                          0.63  0.000 0.000  0.000  0.000
|  TCPIP    R           62                                0.000 0.000  0.000  0.000
|  TPNSREP  R           85                                0.000 0.000  0.000  0.000
|  TSOREP   R           51                                8.283 0.000  6.840  6.840
|  WCSQBRK  R          0.0                                0.000 0.000  0.000  0.000
|  WCSQMSTR R           56                                0.000 0.000  0.000  0.000
|  WLM      R           97                                0.000 0.000  0.000  0.000
|  WPDB2    R           17                                0.000 0.000  0.000  0.000
|  WPOMV    R          0.0                                0.233 0.002  0.509  0.511
|  WT9STDM  R          0.0                                0.000 0.000  0.000  0.000
|  WT9STDMS R          0.0                                0.000 0.000  0.000  0.000
```

```
WT9ST00  R          0.0                                0.000 0.000  0.000  0.000
WT9ST00S R          0.0                                0.000 0.000  0.000  0.000
XCFAS    R           92                                0.000 0.000  0.000  0.000
```

RMF Workload Activity in WLM goal mode

The following illustrates a couple of sections from our RMF *Workload Activity* report in WLM goal mode. This report is based on a 15-minute interval. Highlighted on the report you see 94.3% of our CICS transactions are completing in 0.5 seconds, and our CICS workload is processing 1941.35 transactions per second.

```
1                                        W O R K L O A D   A C T I V I T Y
                                                                                                                PAGE   6
         z/OS V1R8              SYSPLEX UTCPLXJ8          START 01/26/2007-10.30.00 INTERVAL 000.14.59   MODE = GOAL
                             CONVERTED TO z/OS V1R8 RMF     END   01/26/2007-10.44.59

                                       POLICY ACTIVATION DATE/TIME 10/25/2006 13.29.47

   REPORT BY: POLICY=WLMPOL01   WORKLOAD=CICS       SERVICE CLASS=CICS       RESOURCE GROUP=*NONE      PERIOD=1 IMPORTANCE=2
                                                    CRITICAL     =NONE

   TRANSACTIONS   TRANS-TIME HHH.MM.SS.TTT
   AVG      0.00  ACTUAL                22
   MPL      0.00  EXECUTION             18
   ENDED  346638  QUEUED                 0
   END/S  385.16  R/S AFFIN              0
   #SWAPS      0  INELIGIBLE             0
   EXCTD  473510  CONVERSION             0
   AVG ENC  0.00  STD DEV              112
   REM ENC  0.00
   MS ENC   0.00

              RESP  ------------------------------- STATE SAMPLES BREAKDOWN (%) -------------------------------  ------STATE------
   SUB    P   TIME  --ACTIVE-- READY IDLE  ----------------------------WAITING FOR-----------------------------  SWITCHED SAMPL(%)
   TYPE        (%)   SUB  APPL             CONV  I/O LOCK MISC PROD LTCH                                        LOCAL SYSPL REMOT
   CICS  BTE  77.6   3.3   0.0   0.3  0.1  95.8  0.4  0.0  0.0  0.1  0.0                                         88.2   8.0   0.0
   CICS  EXE  28.8  60.6   0.0   6.9  0.0   0.0 29.8  0.0  0.2  2.5  0.0                                          7.2  42.5   0.0
   DB2   BTE   0.0   0.0   0.0   0.0  0.0   0.0  0.0  0.0  0.0  0.0  0.0                                          0.0   0.0   0.0
   DB2   EXE  19.3  86.2   0.0   0.0  0.0   0.0  0.4  0.6 12.6  0.0  0.2                                          0.0   0.0   0.0
   IMS   BTE   0.9  98.9   0.0   0.0  0.0   0.0  0.0  1.1  0.0  0.0  0.0                                          0.0   0.0   0.0
   IMS   EXE   5.4  90.9   0.0   0.0  0.0   0.0  0.0  9.1  0.0  0.0  0.0                                          0.0   0.0   0.0
   SMS   BTE   0.0   0.0   0.0   0.0  0.0   0.0  0.0  0.0  0.0  0.0  0.0                                          0.0   0.0   0.0
   SMS   EXE  31.6  12.1   0.0   0.0  0.0   0.0 55.4 32.5  0.0  0.0  0.0                                          0.0   0.0   0.0

   GOAL: RESPONSE TIME 000.00.00.600 FOR  80%

            RESPONSE TIME EX   PERF
   SYSTEM      ACTUAL%    VEL% INDX

   *ALL           100      N/A  0.5
   JE0            100      N/A  0.5
   J80            100      N/A  0.5
   J90            100      N/A  0.5
   Z0             100      N/A  0.5

1                                        W O R K L O A D   A C T I V I T Y
                                                                                                                PAGE   6
         z/OS V1R8              SYSPLEX UTCPLXJ8          START 01/26/2007-10.30.00 INTERVAL 000.14.59   MODE = GOAL
                             CONVERTED TO z/OS V1R8 RMF     END   01/26/2007-10.44.59

                                       POLICY ACTIVATION DATE/TIME 10/25/2006 13.29.47

                                    ----------RESPONSE TIME DISTRIBUTION----------
      ----TIME----    --NUMBER OF TRANSACTIONS--    -------PERCENT-------  0    10   20   30   40   50   60   70   80   90   100
      HH.MM.SS.TTT    CUM TOTAL       IN BUCKET    CUM TOTAL   IN BUCKET  |....|....|....|....|....|....|....|....|....|....|
   <  00.00.00.300          344K            344K        99.3        99.3  >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   <= 00.00.00.360          345K            741         99.5         0.2  >
   <= 00.00.00.420          345K            461          100         0.1  >
   <= 00.00.00.480          346K            324          100         0.1  >
   <= 00.00.00.540          346K            229          100         0.1  >
   <= 00.00.00.600          346K            151          100         0.0  >
   <= 00.00.00.660          346K            128          100         0.0  >
   <= 00.00.00.720          346K             86          100         0.0  >
   <= 00.00.00.780          346K             52          100         0.0  >
   <= 00.00.00.840          346K             36          100         0.0  >
   <= 00.00.00.900          346K             39          100         0.0  >
   <= 00.00.01.200          346K            102          100         0.0  >
```

```
|     <= 00.00.02.400          347K             95          100         0.0  >
|     >  00.00.02.400          347K            134          100         0.0  >
|     ================================================================================================== WORKLOAD
|
|     REPORT BY: POLICY=WLMPOL01   WORKLOAD=CICS
|                                       cics workload
|
|     TRANSACTIONS   TRANS-TIME HHH.MM.SS.TTT  --DASD I/O--  ---SERVICE----  SERVICE TIMES  ---APPL %---  PAGE-IN RATES   ---STORAGE---
|     AVG     41.00  ACTUAL               143  SSCHRT  2536  IOC     23626K  CPU   2079.9   CP    297.69  SINGLE     0.0  AVG  17179.02
|     MPL     41.00  EXECUTION             62  RESP     1.7  CPU    341640K  SRB    607.7   AAPCP   3.00  BLOCK      0.0  TOT  704281.3
|     ENDED  496115  QUEUED                 0  CONN     0.7  MSO      1444M  RCT      0.0   IIPCP   0.28  SHARED     0.0  CEN  704281.3
|     END/S  551.25  R/S AFFIN              0  DISC     0.8  SRB     97488K  IIT     13.6                 HSP        0.0  EXP      0.00
|     #SWAPS      0  INELIGIBLE             0  Q+PEND   0.3  TOT      1907M  HST      0.0   AAP     0.97  HSP MISS   0.0
|     EXCTD  524444  CONVERSION             0  IOSQ     0.0  /SEC     2119K  AAP      8.7   IIP     1.47  EXP SNGL   0.0  SHR    227.98
|     AVG ENC  0.00  STD DEV           1.130                                  IIP     13.3                 EXP BLK    0.0
|     REM ENC  0.00                                          ABSRPTN    52K                               EXP SHR    0.0
|     MS ENC   0.00                                          TRX SERV   52K
|
```

Appendix B. Availability of our test reports

We publish our test reports twice a year, every June and December. Our December edition covers our initial test experiences with a new z/OS release, including migration. Our June edition is the final edition for that z/OS release; it is cumulative, building upon the December edition with any new test experiences we've encountered since then. We freeze the June edition and begin anew with the next release in December.

You can access our test reports on the Internet or on IBM Softcopy collections.

Availability on the Internet: You can view, download, and print the most current edition of our test report from our System z Platform Test Web site at www.ibm.com/servers/eserver/zseries/zos/integtst/.

You can also find our test reports on the z/OS Internet Library at www.ibm.com/servers/eserver/zseries/zos/bkserv/.

Each edition is available in the following formats:

- IBM BookManager® BOOK format

  On the Web, BookManager documents are served as HTML via IBM LibraryServer. You can use your Web browser (no plug-in or other applications are needed) to view, search, and print selected topics. You can also download individual BOOK files and access them locally using the IBM Softcopy Reader or IBM Library Reader™. You can get the Softcopy Reader or Library reader from the IBM Softcopy Web site at www.ibm.com/servers/eserver/zseries/softcopy/.
- Adobe® Portable Document Format (PDF)

  PDF documents require the Adobe Reader to view and print. Your Web browser can invoke the Adobe Reader to work with PDF files online. You can also download PDF files and access them locally using the Adobe Reader. You can get the Adobe Reader from www.adobe.com/products/acrobat/readstep.html.

Softcopy availability: BookMaster® BOOK and Adobe PDF versions of our test reports are included in the z/OS softcopy collections on CD-ROM and DVD. For more information about softcopy deliverables and tools, visit the IBM Softcopy Web site.

A note about the currency of our softcopy editions

Because we produce our test reports twice a year, June and December, we cannot meet the production deadline for the softcopy collections that coincide with the product's GA release and the softcopy collection refresh date six months later. Therefore, there is normally a one-edition lag between the release of our latest test report edition and the softcopy collection in which it is included. That is, the test report that appears in any given softcopy collection is normally one edition behind the most current edition available on the Web.

Other related publications: From our Web site, you can also access other related publications, including our companion publication, *z/OS V1R8.0 System z Parallel Sysplex Recovery*, GA22-7286.

Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Vol I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

z/OS information

z/OS information is accessible using screen readers with the BookServer/Library Server versions of z/OS books in the Internet library at:

`www.ibm.com/servers/eserver/zseries/zos/bkserv/`

Notices

This information was developed for products and services offered in the USA.

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

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

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

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

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

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

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
2455 South Road
Poughkeepsie, NY 12601-5400
USA

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

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

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

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

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

IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

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

AIX
BatchPipes
BookManager
BookMaster
CICS
DB2
DB2 Connect
DFSMS/MVS
DFSMSrmm
Domino
DRDA
DS6000
DS8000
Enterprise Storage Server
ESCON
eServer
FICON
FlashCopy
IBM
ibm.com
IBMLink
IMS
Language Environment
Library Reader
MQSeries
MVS
OMEGAMON
OS/390
Parallel Sysplex
QMF
RACF
Rational
Redbooks
REXX
RMF
RS/6000
S/390
SupportPac
Sysplex Timer
System Storage
System x
System z
System z9
Tivoli
Tivoli Enterprise
Tivoli Enterprise Console
TME
TotalStorage
VTAM
WebSphere
z/OS
z/OS.e
z/VM
zSeries
z9

The following terms are trademarks of other companies:

Adobe is a registered trademark of Adobe Systems Incorporated in the United States, and/or other countries.

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

Windows is a trademark of Microsoft Corporation in the United States, other countries, or both.

Intel is a registered trademark of Intel Corporation in the United States and other countries.

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

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

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

Index

Readers' Comments — We'd Like to Hear from You

z/OS
System z Platform Test Report for z/OS and Linux Virtual Servers
Version 1 Release 9

Publication No. SA22-7997-06

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Comments:

Thank you for your support.

Submit your comments using one of these channels:

- Send your comments to the address on the reverse side of this form.

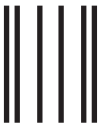
If you would like a response from IBM, please fill in the following information:

Name	Address
Company or Organization	
Phone No.	E-mail address

Readers' Comments — We'd Like to Hear from You
SA22-7997-06

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Department B6ZH, Mail Station P350
2455 South Road
Poughkeepsie, NY
12601-5400

Fold and Tape | **Please do not staple** | Fold and Tape

Cut or Fold
Along Line

Printed in USA

SA22-7997-06