

z/OS



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

Version 1 Release 6

z/OS



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

Version 1 Release 6

Note!

Before using this information and the products it supports, be sure to read the general information under "Notices" on page 325.

Second Edition, June 2005

This is a major revision of SA22-7997-00.

This edition applies to Parallel Sysplex environment function that includes data sharing and parallelism. Parallel Sysplex uses the OS/390 (5647-A01), z/OS (5694-A01), or z/OS.e (5655-G52) 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(BIXLER)

Internet e-mail: bixler@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.

© Copyright International Business Machines Corporation 2001, 2005. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Opening remarks

A message from our team

We changed our title from the *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's and Linux on zSeries' place in the enterprise. To reflect that focus, we changed our title to be the **zSeries® Platform Test Report** .

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, please see "How to send your comments" on page xxii.

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	Luis Cruz	Sue Marcotte
Loraine Arnold	Martha DeMarco	Tammy McAllister
Ozan Baran	Tony DiLorenzo	James Mitchell
Ryan Bartoe	Bob Fantom	Bob Muenkel
Duane Beyer	Nancy Finn	Elaine Murphy
Jeff Bixler	Bobby Gardinor	Jim Rossi
Muriel Bixler	Kieron Hinds	Tom Sirc
Dave Buehl	Gerry Hirons	Karen Smolar
Jon Burke	Joan Kelley	Jeff Stokes
Alex Caraballo	Frank LeFevre	Jim Stutzman
Phil Chan	Parul Lewicke	Lissette Toledo
John Corry	Fred Lates	Ashwin Venkatraman
Don Costello	Al Lease	Jin Xiong
Vince Crose	Scott Loveland	Jessie Yu

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 toward the end of the product development cycle, just before each new software release becomes generally available (GA), we cannot meet the production deadline for the softcopy collections that coincide with the product's GA release. 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 zSeries Platform Test Report Web site at:

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

Contents

Opening remarks	iii
Important—Currency of the softcopy edition	v
Figures	xv
Tables	xvii
About this document	xix
An overview of Integration Test	xix
Our mission and objectives	xix
Our test environment.	xx
Who should read this document.	xx
How to use this document	xx
How to find the Parallel Sysplex Test Report	xx
Where to find more information	xxi
Using LookAt to look up message explanations	xxi
How to send your comments	xxii
Summary of changes	xxiii

Part 1. Parallel Sysplex 1

Chapter 1. About our Parallel Sysplex environment.	5
Overview of our Parallel Sysplex environment	5
Our Parallel Sysplex hardware configuration	5
Overview of our hardware configuration	5
Hardware configuration details.	7
Our Parallel Sysplex software configuration	16
Overview of our software configuration	16
About our naming conventions	18
Our Integrated Cryptographic Service Facility (ICSF) configuration	18
Our networking configuration	19
Our VTAM configuration	19
Our workloads	20
Base system workloads.	20
Application enablement workloads	21
Networking workloads	24
Database product workloads	25
Chapter 2. Migrating to and using z/OS	29
Overview	29
Migrating to z/OS V1R6	29
z/OS V1R6 base migration experiences	29
Defining greater than 16 CPs per z/OS image	31
Migrating to z/OS.e V1R6	31
z/OS.e V1R6 base migration experiences	31
Other experiences with z/OS.e V1R6.	35
Migrating to z/OS V1R5	35
z/OS V1R5 base migration experiences	35
Using DFSMS enhanced data integrity for sequential data sets	37
Using the new XCF REALLOCATE process	38
Migrating to z/OS.e V1R5	48

	z/OS.e V1R5 base migration experiences	48
	Other experiences with z/OS.e V1R5.	51
	Using the IBM Health Checker for z/OS and Sysplex	51
	z/OS performance.	52
	Chapter 3. Using zSeries Application Assist Processors (zAAPs)	53
	Prerequisites for zAPP	53
	Subsystems and applications using SDK 1.4 that exploit zAAPs	53
	Setting up zAAP	53
	Configuring zAAPs	54
	Monitoring zAAP utilization	55
	Preparing our workloads to exercise the zAAP feature	56
	Chapter 4. Migrating to CICS TS Version 2 Release 3	59
I	Overview of migrating to CICS TS 2.3	59
I	Performing the migration to CICS TS 2.3	60
I	Preparing for migration	60
I	Migrating CICSplex SM.	61
I	Migrating the CASSs	61
I	Migrating the CMASs	62
I	Migrating the MASs	63
I	Experiences with migrating to CICS TS 2.3	63
	Chapter 5. Migrating to DB2 Version 8	65
I	Migration considerations	65
I	Premigration activities	66
I	Migrating the first member to compatibility mode	69
I	DB2 V7 and V8 coexistence issues	77
I	Migrating the remaining members to compatibility mode	77
I	Migrating to new function mode	82
I	Preparing for new function mode	82
I	Enabling new function mode	86
I	Running in new function mode	88
I	Verifying the installation using the sample applications	89
	Chapter 6. Migrating to IMS Version 9	93
I	Migrating to the integrated IMS Connect	95
I	Migrating to IRLM Version 2 Release 2	96
	Chapter 7. Implementing the IMS Common Service Layer and the Single Point of Control	97
	Setting up the Common Service Layer	97
	Steps for setting up the CSL	97
	Our CSL and SPOC configuration	100
	IMS performance considerations for CSL	101
	Setting up the single point of control	102
	Steps for setting up the single point of control	102
	Steps for setting up DB2 Control Center for the IMS SPOC	103
	Chapter 8. Parallel Sysplex automation	109
	Our early experiences with automation.	109
	Automation with msys for Operations	109
I	Migrating to System Automation for OS/390 Version 2 Release 3	109
	Using SA OS/390	110
	Using the DRAIN and ENABLE subcommands.	110
	Refreshing the automation manager.	110

	Turning off the automation flag for a resource	112
	Chapter 9. Testing SPE Console Restructure (APAR OA09229)	117
<hr/>		
	Part 2. Networking and application enablement.	119
	Chapter 10. About our networking and application enablement environment	123
	Our networking and application enablement configuration	123
	Our Ethernet LAN configuration	124
	Our ATM configuration.	125
	Our ipV6 Environment Configuration	125
	Our token ring LAN configuration.	127
	Comparing the network file systems.	132
	Networking and application enablement workloads	132
	Enabling NFS recovery for system outages	133
	Setting up the NFS environment for ARM and DVIPA	133
	Chapter 11. Using z/OS UNIX System Services	137
	z/OS UNIX enhancements in z/OS V1R5.	137
	Remounting a shared HFS	137
	Mounting file systems using symbolic links	137
	Creating directories during z/OS UNIX initialization	138
	Temporary file system (TFS) enhancements.	141
	z/OS UNIX enhancements in z/OS V1R6.	145
	Using multipliers with BPXPRMxx parameters	146
	Using the superkill option	146
	Using wildcard characters in the automove system list (SYSLIST).	148
	Using the clear and uptime shell commands	149
	Enhanced latch contention detection	150
	Shells and utilities support for 64-bit virtual addressing.	151
	Using distributed BRLM	159
	Using ISHELL enhancements	161
	Using the hierarchical file system (HFS)	164
	Automount enhancement for HFS to zSeries file system (zFS) migration	164
	Using the zSeries file system (zFS)	165
	zFS enhancements in z/OS V1R6	165
	HANGBREAK, zFS modify console command	168
	Issuing the su command and changing TSO identity.	169
	Removing additional diagnostic data collection from OMVS CTRACE LOCK processing	169
	Chapter 12. Using the IBM HTTP Server	171
	Using gskkyman support for storing a PKCS #7 file with a chain of certificates	171
	Chapter 13. Using LDAP Server	173
	Overview of our LDAP configuration.	173
	Setting up the LDAP server for RACF change logging	174
	Activating change notification in RACF.	175
	Setting up the GDBM backend for the LDAP server	175
	Testing the change logging function and the GDBM database	177
	Using the z/OS LDAP client with the Windows 2000 Active Directory service	183
	Using LDAP with Kerberos authentication	184
	Problems we experienced with our workload	184
	Setting up SSL client and server authentication between z/OS LDAP V1R6 server/client and Sun ONE Directory Server 5.2 server/client.	186

	Setting up SSL client and server authentication between z/OS LDAP V1R6	
	server/client and IBM Tivoli Directory Server 5.2 server/client	192
	LDAP Server enhancements in z/OS V1R6	196
	LDAP migration to z/OS V1R6.	196
	Setting up a peer-to-peer replication network between an IBM Tivoli	
	Directory Server 5.2 and a z/OS LDAP Server	197
	Using DB2 restart/recovery function.	203
	Migrating to DB2 V8	204
	Using alias support	205
	Using the enhanced LDAP configuration utility (LDAPCNF).	206
	Using change logging with TDBM	207
	Chapter 14. Using Kerberos (Network Authentication Service)	211
	Setting up a Kerberos peer trust relationship between z/OS and Windows 2000	211
	Enabling the peer trust relationship on z/OS.	211
	Testing the peer trust relationship	212
	Network Authentication Service (NAS) enhancements in z/OS V1R6.	213
	Accessing SYS1.SIEALNKE	213
	FTP with Kerberos	214
	Where to find more information	214
	FTP server enablement for Kerberos	214
	Configuring a Linux workstation for Kerberos	216
	Creating the ftp.data file for the z/OS client ftp user	216
	Testing FTP with Kerberos	217
	Problems encountered	217
	Working with Kerberos principals in RACF	217
	Chapter 15. Using the IBM WebSphere Business Integration family of	
	products.	219
	Using WebSphere MQ shared queues and coupling facility structures	219
	Our queue sharing group configuration	219
	Our coupling facility structure configuration	219
	Testing the recovery behavior of the queue managers and coupling facility	
	structures	220
	Implementing WebSphere MQ shared channels in a distributed-queuing	
	management environment	222
	Our shared channel configuration	223
	Testing shared channel recovery	224
	Using WebSphere Business Integration Message Broker	226
	Testing WMQI V2.1 on DB2 V8	226
	Setting the _BPXK_MDUMP environment variable to write broker core	
	dumps to MVS data sets	226
	Resolving a EC6-FF01 abend in the broker.	228
	Migrating WebSphere MQ Integrator V2.1 to WebSphere Business	
	Integration Message Broker V5.0	228
	Applying WBIMB V5.0 Fix Pack 02 and Fix Pack 03.	229
	Some useful WBIMB Web sites	229
	Chapter 16. Using IBM WebSphere Application Server for z/OS	231
	About our z/OS V1R6 test environment running WebSphere Application Server	231
	Our z/OS V1R6 WebSphere test environment	231
	Other changes and updates to our WebSphere test environment	235
	Migrating WebSphere Application Server for z/OS JDBC from DB2 V7 to	
	DB2 V8	235
	Using DB2 UDB JCC Connectors	235
	Migrating to CICS Transaction Gateway Connector V5.1	236

Enabling Global Security and SSL on WebSphere Application Server for z/OS	236
Using the WebSphere Application Server for z/OS 5.x plug-in for HTTP Server and Sysplex Distributor with our WebSphere Application Server for z/OS J2EE Servers	238
Where to find more information	248
Specific documentation we used	248
Chapter 17. Using EIM authentication	251
Client authentication using digital certificates	251
Resolving problems during our testing	251
Testing the client authentication using digital certificates	252
Kerberos authentication	252
Clearing up a documentation inaccuracy	253
Testing the Kerberos authentication	253
CRAM-MD5 password protection	254
EIM enhancements in z/OS V1R6	254
x.509 certificate registries	254

Part 3. Linux virtual servers 259

Chapter 18. About our Linux virtual server environment 261

Chapter 19. Cloning Linux images on z/VM 5.1 263

Preparing the VM environment for the cloning system	264
Adding the FLASHCOPY command to the "Z" class.	264
Defining VM userid "USER" and common DASD	264
Defining the key files on common DASD	264
Including the LTICPRO directory profile	265
Defining the LTICxxx directory entry.	266
Setting up the LTICxxx system	266
Setting the IP and HOSTNAME on the LTICxxxx guest sytem	266
Verifying the setup	267

Chapter 20. Establishing security in a heterogeneous Linux server environment 269

Planning for our Linux on zSeries environment.	269
Linux on zSeries network configuration	270
Linux on zSeries middleware environment	273
Existing environment	273
New middleware	273
Installing WebSphere Application Server and WebSphere Application Server Network Deployment V5.1	273
Web Servers for WebSphere Application Server and zSeries Hardware Cryptographic Accleration.	274
Configuring DB2 V8.1 clients on Linux on zSeries to a z/OS DB2 backend	278
Setting SSL tunneling on in the WebSphere Application Server Edge Component Caching Proxy V5.1	279
Configuring WebSphere Application Server ND Edge Component Load Balancer V5.1	280
Defining Samba on Red Hat Enterprise Linux 3 Update 4.	281
Installing and running Domino Mail Server V6.5.4 on Linux on zSeries	282
Open source security products	286
Changing the placement of Hogwash	286
iptables	286

	Defining the rules for the firewall between between Public LAN and	
	VLAN674.	286
	Defining the rules for the firewall between VLAN673 and VLAN672	287
	IBM security products	290
	Planning and installing Tivoli Risk Manager (TRM) Host IDs	290
	Authenticating and authorizing Web transactions using Tivoli Access	
	Manager and TAM WebSeal.	295
	Authenticating Linux users using RACF and LDAP on z/OS	296
	Independent service vendor (ISV) security products	297
	Installing TrendMicro's ScanMail	297
	Installing TrendMicro's ServerProtect	298
	Security testing	301
	Security: Next steps	301
	Chapter 21. Future Linux on zSeries projects	303
	Migration	303
	High availability	303
	Where to find more information	303
	Appendix A. Some of our parmlib members.	305
	Appendix B. Some of our RMF reports.	307
	RMF Monitor I post processor summary report.	307
	RMF Monitor III online sysplex summary report	307
	RMF workload activity report in WLM goal mode	308
	On/Off Capacity on Demand Testing	310
	Appendix C. Some of our Linux for zSeries samples, scripts and EXECs	313
	Files on our FTP server	313
	IticlPsetup	313
	ip.list	313
	Files on our USER 194 disk	314
	PROFILE EXEC	314
	WELCOME EXEC	314
	Files on our USER 195 disk	315
	DISKCOPY EXEC	315
	DISKCOPY LIST.	316
	DISTRO EXEC	316
	DISKCOPY LIST.	316
	IPDATA EXEC.	317
	IPDATA LIST	317
	LOADRDR EXEC	318
	Appendix D. Availability of our test reports	319
	Appendix E. Useful Web sites	321
	IBM Web sites	321
	Other Web sites	322
	Appendix F. Accessibility	323
	Using assistive technologies	323
	Keyboard navigation of the user interface.	323
	z/OS information	323
	Notices	325
	Trademarks.	327

Index 329

Figures

1.	Our sysplex hardware configuration	6
2.	Our coupling facility channel configuration.	11
3.	Our sysplex software configuration	17
4.	Our VTAM configuration	19
5.	Example of the image profile for our Z2 image with two zAAPs defined.	54
6.	Our CICS TS 2.3 and CPSM 2.3 configuration	60
7.	DSNTIPA1	67
8.	DSNTIPP2	68
9.	Tailored CLISTS placed in DB2.DB2810.DBD1.SDSNSAMP and DB2.DB2810.NEW.SDSNTEMP	69
10.	Output from query to find packages that will be invalidated when migrating to DB2 Version 8	71
11.	DISPLAY GROUP command	73
12.	Message DSNU777I displays CATMAINT progress	74
13.	SPUFI is not available for use on DB2 Version 7 members after the execution of DSNTIJSJG	75
14.	Executing DSNTINST in preparation for migrating the next member of the data sharing group	78
15.	DSNTIPP2 pop-up screen	78
16.	DSNTIPT - Data Set Names Panel 1	79
17.	Tailored migration JCL placed in DB2.DB2810.DBG1.SDSNSAMP	80
18.	DBG1 started in compatibility mode	81
19.	All members now in compatibility mode	81
20.	Executing DSNTINST in preparation for enabling-new-function-mode.	82
21.	DSNTIP00 panel	83
22.	Image copy data set allocations on panel DSNTIP01	83
23.	DSNT470I Warning message, only one volume was specified	84
24.	Message DSNT488I displayed on panel DSNTIP02	84
25.	DSNT478I beginning data set output	85
26.	DSNT489I CLIST editing	85
27.	Screen showing completion of the preparation before enabling Version 8 new function mode	86
28.	The DISPLAY GROUP command shows the data sharing group is now in new function mode	88
29.	Our IMS CSL and SPOC configuration	101
30.	Example of the Control Center Add System dialog	103
31.	Example of the Command Center initial setup.	104
32.	Example of issuing an IMS command to IMSplex member IMSC	105
33.	Example of the response to an IMS command that was issued to IMSplex member IMSC	106
34.	Example of issuing an IMS command to all members of the IMSplex	107
35.	Example of the response to an IMS command that was issued to all members of the IMSplex	108
36.	Example of the INGAMS dialog	111
37.	Example of the Refresh Configuration dialog	112
38.	Example display from the DS LDAP* command	113
39.	Example of the automation settings dialog for the LDAPSRV server (automation flag is on)	114
40.	Example of the automation settings dialog for the LDAPSRV server (automation flag is off)	115
41.	Our networking and application enablement configuration	123
42.	Our token-ring LAN A.	129
43.	Our token-ring LAN B	130
44.	Our token-ring LAN C	131
45.	NFS configuration	134
46.	Overview of our LDAP configuration	173
47.	Our WebSphere for z/OS V5.1 configuration	234
48.	Servers in one system of our WebSphere Application Server for z/OS P1 Cell (production)	239
49.	One J2EE Server Cluster (WSP1S1) in our WebSphere Application Server for z/OS P1 Cell (production)	247
50.	Linux on zSeries network configuration.	271
51.	ServerProtect's Scan Complete display.	300
52.	ServerProtect's "Virus Logs" display.	301

53.	Example RMF Monitor I post processor summary report	307
54.	Example RMF Monitor III online sysplex summary report.	308
55.	Example RMF workload activity report in WLM goal mode	309
I 56.	Example RMF workload activity report in WLM goal mode	311

Tables

1.	Parallel Sysplex planning library publications	xxi
2.	Our mainframe servers	7
3.	Our coupling facilities.	9
4.	CFs and CFLEVELS.	12
5.	Other sysplex hardware configuration details	12
6.	Our production OLTP application groups	17
7.	Summary of our workloads	20
8.	Our high-level migration process for z/OS V1R6	30
9.	Our high-level migration process for z/OS.e V1R6	32
10.	Our high-level migration process for z/OS V1R5	35
11.	Summary of the modes of operation for the DFSMS enhanced data integrity function for sequential data sets.	37
12.	Our high-level migration process for z/OS.e V1R5	48
13.	DB2 system table spaces and whether or not new function mode has been enabled yet.	87
14.	Character Parameter Limit Multipliers	146
15.	WebSphere MQ APARs.	221
16.	Middleware compatibility matrix.	269
17.	Summary of our parmlib changes for z/OS V1R5 and z/OS.e V1R5.	305
18.	Available year-end editions of our test report	320
19.	Some IBM Web sites that we reference	321
20.	Other Web sites that we reference	322

About this document

This document is a test report written from the perspective of a system programmer. The IBM zSeries 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 zSeries customer 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 Integration Test

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 recreatable 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 e-business enterprise—just like yours.

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

- “Our Parallel Sysplex hardware configuration” on page 5
- “Our Parallel Sysplex software configuration” on page 16
- “Our networking and application enablement configuration” on page 123
- “Our workloads” on page 20

Who should read this document

System programmers should use this book 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 knowledge of MVS and Parallel Sysplex concepts and terminology and at least a basic level of experience with installing and managing the z/OS or OS/390 operating system, subsystems, network products, and other related software. See “Where to find more information” on page xxi.

How to use this document

Use this document as a companion to—*never a replacement for*—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 the Parallel Sysplex Test Report

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 D, “Availability of our test reports,” on page 319 for other ways to access our test reports.

We have traditionally published our test report on a quarterly basis where each quarterly edition was cumulative for the current year. At the end of each year, we freeze the content in our last edition; we then begin with a new test report the

following year. The most recent quarterly edition as well as all of the previous year-end editions are available on our Web site.

In 2003, our publication schedule changed from our traditional quarterly cycle as a result of the change in the development cycle for annual z/OS releases. Keep an eye on our Web site for announcements about the availability of new editions of our test report. In any event, the contents of our test reports remain cumulative for any given year.

We also have a companion publication, *OS/390 Parallel Sysplex Recovery*, GA22-7286-00, which documents the Parallel Sysplex recovery scenarios we've executed in our test environment, including operating system, subsystem, and coupling facility recovery. We describe how to be prepared for potential problems in a Parallel Sysplex, what the indicators are to let you know that a problem exists, and what actions to take to recover.

Note: The recovery book was written in the OS/390 V2R4 time frame; however, many of the recovery concepts that we discuss still apply to later releases of OS/390 and 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
<i>z/OS Parallel Sysplex Overview</i>	SA22-7661
<i>z/OS MVS Setting Up a Sysplex</i>	SA22-7625
<i>z/OS Parallel Sysplex Application Migration</i>	SA22-7662
<i>z/OS and z/OS.e Planning for Installation</i>	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/psol/
- See the Parallel Sysplex Customization Wizard at:
www.ibm.com/servers/eserver/zseries/zos/wizards/parallel/
- See the z/OS Managed System Infrastructure (msys) for Operations Web site at:
www.ibm.com/servers/eserver/zseries/msys/msysops/

Using LookAt to look up message explanations

LookAt is an online facility that lets you look up explanations for most of the IBM® messages you encounter, as well as for some system abends and codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can use LookAt from the following locations to find IBM message explanations for z/OS elements and features, z/VM®, VSE/ESA™, and Clusters for AIX® and Linux:

- The Internet. You can access IBM message explanations directly from the LookAt Web site at <http://www.ibm.com/eserver/zseries/zos/bkserv/lookat/>.

- Your z/OS TSO/E host system. You can install code on your z/OS or z/OS.e systems to access IBM message explanations, using LookAt from a TSO/E command line (for example, TSO/E prompt, ISPF, or z/OS UNIX[®] System Services).
- Your Microsoft[®] Windows[®] workstation. You can install code to access IBM message explanations on the *z/OS Collection* (SK3T-4269), using LookAt from a Microsoft Windows command prompt (also known as the DOS command line).
- Your wireless handheld device. You can use the LookAt Mobile Edition with a handheld device that has wireless access and an Internet browser (for example, Internet Explorer for Pocket PCs, Blazer, or Eudora for Palm OS, or Opera for Linux handheld devices). Link to the LookAt Mobile Edition from the LookAt Web site.

You can obtain code to install LookAt on your host system or Microsoft Windows workstation from a disk on your *z/OS Collection* (SK3T-4269), or from the LookAt Web site (click **Download**, and select the platform, release, collection, and location that suit your needs). More information is available in the LOOKAT.ME files available during the download process.

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:

- dilorenz@us.ibm.com for z/OS questions
- jinxiang@us.ibm.com for Linux on zSeries questions

or use the contact form on our Web site at:

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

You can also submit the Readers' Comments form located at the end of this 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 heading or page number).

Summary of changes

We periodically update our test report with new information and experiences. If the edition you are currently reading is more than a few months old, you may want to check whether a newer edition is available (see “How to find the Parallel Sysplex Test Report” on page xx).

This information below summarizes the changes that we have made to this document.

Summary of changes for SA22-7997-01 June 2005

This document contains information previously presented in SA22-7997-00.

New information

- “CFCC Dispatcher Rewrite testing” on page 11
- “Exploiting 64k cylinder logical volumes” on page 13
- “Testing greater than 16 CPU Support” on page 14
- “Our Integrated Cryptographic Service Facility (ICSF) configuration” on page 18
- “On/Off Capacity On Demand Testing” on page 15
- “z800 Concurrent upgrade testing” on page 15
- Chapter 4, “Migrating to CICS TS Version 2 Release 3,” on page 59
- Chapter 5, “Migrating to DB2 Version 8,” on page 65
- Chapter 6, “Migrating to IMS Version 9,” on page 93
- “Migrating to System Automation for OS/390 Version 2 Release 3” on page 109
- Chapter 9, “Testing SPE Console Restructure (APAR OA09229),” on page 117
- “Issuing the su command and changing TSO identity” on page 169
- “Removing additional diagnostic data collection from OMVS CTRACE LOCK processing” on page 169
- “Setting up SSL client and server authentication between z/OS LDAP V1R6 server/client and Sun ONE Directory Server 5.2 server/client” on page 186
- “Setting up SSL client and server authentication between z/OS LDAP V1R6 server/client and IBM Tivoli Directory Server 5.2 server/client” on page 192
- “Migrating to DB2 V8” on page 204
- “FTP with Kerberos” on page 214
- “Working with Kerberos principals in RACF” on page 217
- “About our z/OS V1R6 test environment running WebSphere Application Server” on page 231
- “Migrating WebSphere Application Server for z/OS JDBC from DB2 V7 to DB2 V8” on page 235
- “Using DB2 UDB JCC Connectors” on page 235
- “Migrating to CICS Transaction Gateway Connector V5.1” on page 236
- “Enabling Global Security and SSL on WebSphere Application Server for z/OS” on page 236
- “Using the WebSphere Application Server for z/OS 5.x plug-in for HTTP Server and Sysplex Distributor with our WebSphere Application Server for z/OS J2EE Servers” on page 238

- Chapter 18, “About our Linux virtual server environment,” on page 261
- Chapter 19, “Cloning Linux images on z/VM 5.1,” on page 263
- Chapter 20, “Establishing security in a heterogeneous Linux server environment,” on page 269
- Chapter 21, “Future Linux on zSeries projects,” on page 303
- “On/Off Capacity on Demand Testing” on page 310

Changed information

- Our sysplex hardware configuration
- “WebSphere MQ workloads” on page 22
- “WebSphere Business Integration Message Broker” on page 23
- “Migrating to System Automation for OS/390 Version 2 Release 3” on page 109

This document contains terminology, maintenance, and editorial changes, including changes to improve consistency and retrievability. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

References to OpenEdition have been replaced with z/OS UNIX System Service or z/OS UNIX.

Summary of changes for SA22-7997-00 September 2004

This document contains information previously presented in SA22-7663-11.

New information

- Enabling NFS recovery for system outages
- Automount enhancement for HFS to zSeries file system (zFS) migration
- Using multipliers with BPXPRMxx parameters
- Using the superkill command
- Added an IPv6 environment equivalent to our IPv4 environment. V1R6 now supports OSPF V3 for IPv6 and IPv6 support for DVIPA and Sysplex Distributor.
- Using wildcard characters in the automove system list (SYSLIST)
- Using the clear and uptime shell commands
- Enhanced latch contention detection
- Using distributed BRLM
- Using ISHELL enhancements
- zFS modify console command
- Using gskkyman support for storing a PKCS #7 file with a chain of certificates
- LDAP migration to z/OS V1R6
- Setting up a peer-to-peer replication network between an IBM Tivoli® Directory Server 5.2 and a z/OS LDAP Server
- Using LDAP DB2® restart/recovery function
- Using LDAP alias support
- Using the enhanced LDAP configuration utility (LDAPCNF)
- Using LDAP change logging with TDBM
- NAS accessing SYS1.SIEALNKE
- EIM enhancements in z/OS V1R6

- Updates to our z/OS V1R5 test environment running WebSphere® Application Server
- Migrating to WebSphere for z/OS V5.X on z/OS V1R6

Changed information

- Our sysplex hardware configuration

This document contains terminology, maintenance, and editorial changes, including changes to improve consistency and retrievability. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Summary of changes

SA22-7663-11

June 2004

This document contains information previously presented in SA22-7663-10.

New information

- XCF REALLOCATE processing
- Using zSeries Application Assist Processors (zAAPs)
- IBM Health Checker for z/OS and Sysplex Version 3
- Migrating to WebSphere Business Integration Message Broker Version 5.0
- Implementing shared channels in a distributed-queuing management (DQM) environment
- Setting up a Kerberos peer trust relationship between z/OS and Windows 2000
- Using the z/OS LDAP client with the Windows 2000 Active Directory service
- Using LDAP with Kerberos authentication

Changed information

- Our sysplex hardware configuration

This document contains terminology, maintenance, and editorial changes, including changes to improve consistency and retrievability. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Summary of changes

for SA22-7663-10

March 2004

This document contains information previously presented in SA22-7663-09.

New information

- Migrating to z/OS V1R5 and z/OS.e V1R5
- Using DFSMS enhanced data integrity for sequential data sets
- Implementing the common service layer (CSL) and single point of control (SPOC) in IMS™ V8
- Using System Automation OS/390 (SA OS/390) V2R2
- Enhancements in z/OS UNIX in z/OS V1R5, including:
 - Remounting a shared HFS
 - Mounting file systems using symbolic links
 - Creating directories during z/OS UNIX initialization
 - Temporary file system (TFS) enhancements

- Setting up the LDAP server for RACF® change logging
- Support for additional bind types for EIM authentication
- Using WebSphere MQ shared queues and coupling facility structures
- Migrating to WebSphere for z/OS V5.0

Changed information

- Overview of our LDAP Server configuration

Deleted information

- System-managed coupling facility structure duplexing
- Verifying use of the cryptographic hardware
- Migrating to IMS Version 8 Release 1
- Using z/OS DFSMSStvs
- Setting up the CICSplex SM Web User Interface
- Using IBM HTTP Server
- LDAP Server enhancements in z/OS V1R4
- Using IMS Connect for z/OS Version 1.2
- Implementing the System SSL started task
- Using PKI Services
- Using WebSphere Studio Workload Simulator in z/OS Integration Test

Although the above information has been deleted from this edition, it continues to be available in our December 2003 edition.

Part 1. Parallel Sysplex

Chapter 1. About our Parallel Sysplex environment.	5
Overview of our Parallel Sysplex environment	5
Our Parallel Sysplex hardware configuration	5
Overview of our hardware configuration	5
Hardware configuration details.	7
Mainframe server details.	7
Coupling facility details	9
Other sysplex hardware details	12
Exploiting 64k cylinder logical volumes	13
Testing greater than 16 CPU Support	14
On/Off Capacity On Demand Testing	15
Our Parallel Sysplex software configuration	16
Overview of our software configuration	16
About our naming conventions	18
Our Integrated Cryptographic Service Facility (ICSF) configuration	18
Our networking configuration	19
Our VTAM configuration	19
Our workloads	20
Base system workloads.	20
Application enablement workloads.	21
Enterprise Identity Mapping (EIM)	21
IBM HTTP Server	21
LDAP Server	21
z/OS UNIX Shelltest (rlogin/telnet)	21
z/OS UNIX Shelltest (TSO)	22
WebSphere Application Server for z/OS.	22
WebSphere MQ workloads	22
Websphere Business Integration Message Broker	23
Networking workloads	24
Database product workloads	25
Database product OLTP workloads	25
Database product batch workloads	26
WebSphere MQ / DB2 bookstore application	26
Chapter 2. Migrating to and using z/OS	29
Overview	29
Migrating to z/OS V1R6	29
z/OS V1R6 base migration experiences.	29
Our high-level migration process for z/OS V1R6.	29
More about our migration activities for z/OS V1R6.	30
Defining greater than 16 CPs per z/OS image	31
Migrating to z/OS.e V1R6	31
z/OS.e V1R6 base migration experiences	31
Our high-level migration process for z/OS.e V1R6	31
More about our migration activities for z/OS.e V1R6	33
Other experiences with z/OS.e V1R6.	35
Migrating to z/OS V1R5	35
z/OS V1R5 base migration experiences.	35
Our high-level migration process for z/OS V1R5.	35
More about our migration activities for z/OS V1R5.	36
Using DFSMS enhanced data integrity for sequential data sets	37
Using the new XCF REALLOCATE process	38
Migrating to z/OS.e V1R5	48

z/OS.e V1R5 base migration experiences	48
Our high-level migration process for z/OS.e V1R5	48
More about our migration activities for z/OS.e V1R5	49
Other experiences with z/OS.e V1R5.	51
Using the IBM Health Checker for z/OS and Sysplex	51
z/OS performance.	52
Chapter 3. Using zSeries Application Assist Processors (zAAPs)	53
Prerequisites for zAPP	53
Subsystems and applications using SDK 1.4 that exploit zAAPs	53
Setting up zAAP	53
Configuring zAAPs	54
Monitoring zAAP utilization	55
Preparing our workloads to exercise the zAAP feature	56
Chapter 4. Migrating to CICS TS Version 2 Release 3	59
Overview of migrating to CICS TS 2.3	59
Performing the migration to CICS TS 2.3	60
Preparing for migration	60
Migrating CICSplex SM.	61
Migrating the CASs	61
Steps for migrating the CASs	61
Migrating the CMASs	62
Steps for migrating the CMASs	62
Migrating the MASs	63
Steps for migrating the MASs	63
Experiences with migrating to CICS TS 2.3	63
Chapter 5. Migrating to DB2 Version 8	65
Migration considerations	65
Premigration activities	66
Migrating the first member to compatibility mode	69
DB2 V7 and V8 coexistence issues	77
Migrating the remaining members to compatibility mode	77
Migrating to new function mode.	82
Preparing for new function mode	82
Enabling new function mode	86
Running in new function mode	88
Verifying the installation using the sample applications	89
Chapter 6. Migrating to IMS Version 9	93
Migrating to the integrated IMS Connect	95
Migrating to IRLM Version 2 Release 2	96
Chapter 7. Implementing the IMS Common Service Layer and the Single Point of Control	97
Setting up the Common Service Layer	97
Steps for setting up the CSL	97
Our CSL and SPOC configuration	100
IMS performance considerations for CSL.	101
Setting up the single point of control	102
Steps for setting up the single point of control	102
Steps for setting up DB2 Control Center for the IMS SPOC	103
Chapter 8. Parallel Sysplex automation	109
Our early experiences with automation.	109

	Automation with msys for Operations	109
I	Migrating to System Automation for OS/390 Version 2 Release 3	109
	Using SA OS/390	110
	Using the DRAIN and ENABLE subcommands.	110
	Refreshing the automation manager.	110
	Turning off the automation flag for a resource	112
I	Chapter 9. Testing SPE Console Restructure (APAR OA09229)	117

The above chapters describe the Parallel Sysplex® aspects of our computing environment.

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 currently run a 14-member Parallel Sysplex that consists of the following:

- Four central processor complexes (CPCs) running z/OS in 14 logical partitions (LPs).

The CPCs consist of the following machine types:

- One IBM @server zSeries 990 (z990) processor
- One IBM @server zSeries 900 (z900) processor
- One IBM @server zSeries 890 (z890) processor
- One IBM @server zSeries 800 (z800) processor

The z/OS images consist of the following:

- Nine production z/OS systems
 - One production z/OS.e system
 - Three test z/OS systems
 - One z/OS system to run TPNS (Our December 1998 edition explains why we run TPNS on a non-production system.)
- Three coupling facilities:
 - One failure-independent coupling facility that runs in a LP on a standalone CPC
 - Two non-failure-independent coupling facilities that run in LPs on two 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.

The remainder of this chapter describes all of the above in more detail.

Outside of the Parallel Sysplex itself, we also have three LPs in which we run the following:

- Two native Linux[®] images
- One z/VM image that hosts 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 6 is a high-level, conceptual view of our Parallel Sysplex hardware configuration. In the figure, broad arrows indicate general connectivity

Parallel Sysplex environment

between processors, coupling facilities, Sysplex Timers, and other I/O devices; they do not depict actual point-to-point connections.

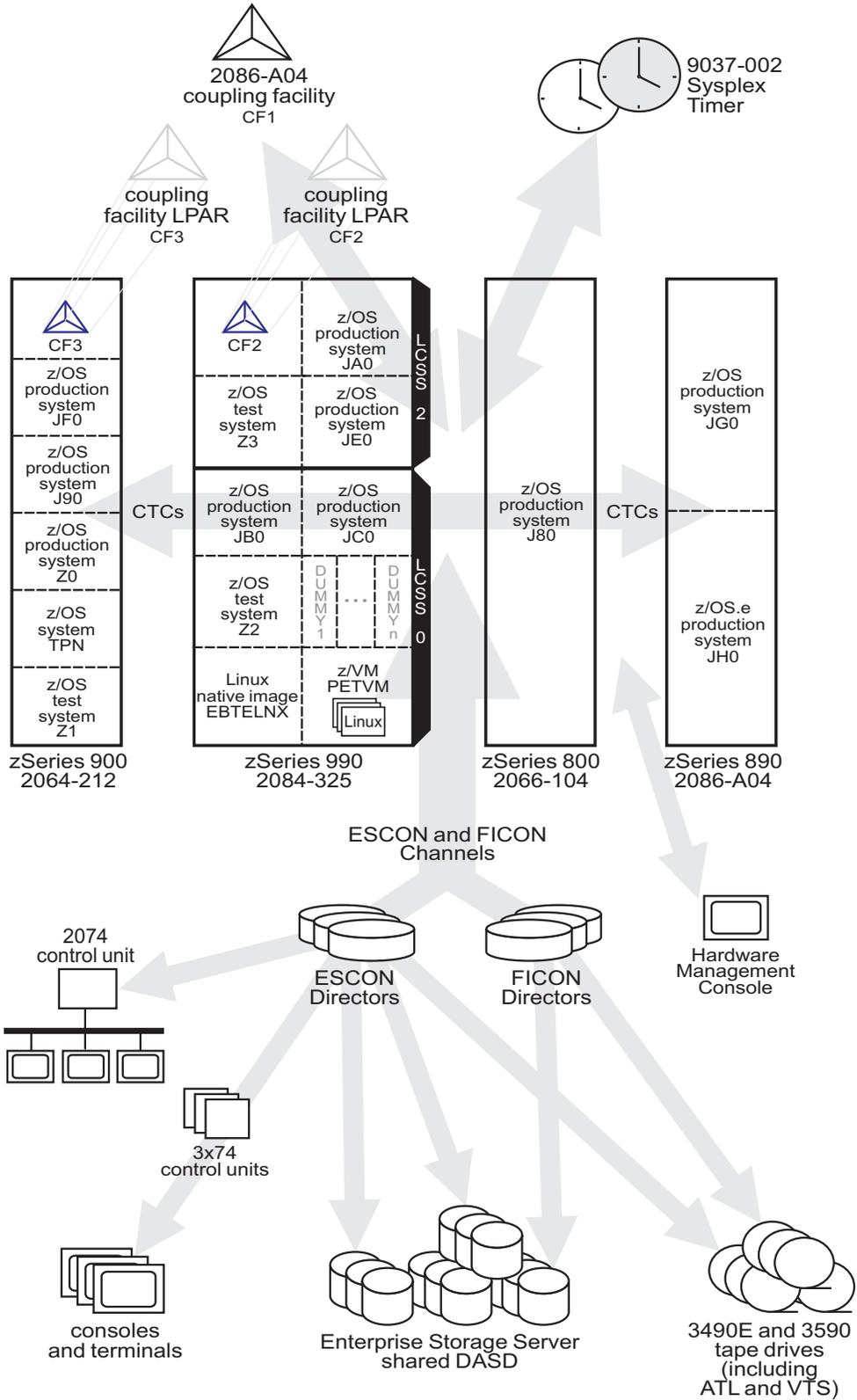


Figure 1. Our sysplex hardware configuration

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 6.

Mainframe server details

Table 2 provides information about the mainframe servers in our sysplex:

Table 2. Our mainframe servers

Server model (Machine type-model)	CPCs CPs	Mode LPs	HSA	Storage: Central Expanded	LCSS	System name, usage Virtual CPs (static, managed) Initial LPAR weight
IBM @server zSeries 800 Model 104 (2066-104)	1 CPC 4 CPs	LPAR 1 LP	160M	7168M		J80 , z/OS production system 4 CPs
(see note 2 below)						
IBM @server zSeries 890 Model A04 (2086-A04)	1 CPC 4 CPs	LPAR mode 2 LPs	1344M	7168M		JG0 , z/OS production system 4 shared CPs weight of 400
(see notes 2 and 3 below)				7168M		JH0 , z/OS.e production system 4 shared CPs weight of 400
IBM @server zSeries 900 Model 212 (2064-212)	1 CPC 16 CPs (4 ICF)	LPAR mode 6 LPs (1 LP is a coupling facility)	256M	5888M		JF0 , z/OS production system 12 shared CPs (8, 4) weight of 285
(see note 1 below)				6144M		J90 , z/OS production system 12 shared CPs (3, 9) weight of 285
				6144M		Z0 , z/OS production system 12 shared CPs (7, 5) weight of 285
				4096M		Z1 , z/OS test system 12 shared CPs (4, 8) weight of 145
				4096M		TPN , z/OS system for TPNS 8 shared CPs
IBM @server zSeries 990 Model 325 (2084-325)	1 CPC 32 CPs (3 ICF, 2 zAAP)	LPAR mode 20 LPs ⁴ (1 LP is a coupling facility)	See note 5.	31G	2	JA0 , z/OS production system 16 shared CPs
				31G	0	JB0 , z/OS production system 16 shared CPs
				31G	0	JC0 , z/OS production system 16 shared CPs
				31G	2	JE0 , z/OS production system 16 shared CPs
				4096M	0	Z2 , z/OS test system 6 shared CPs
				4096M	2	Z3 , z/OS test system 6 shared CPs
				1024M	0	EBTELNX , native Linux image 1 shared CP
				1024M	0	PETVM , z/VM system 2 shared CPs weight of 10

Parallel Sysplex environment

Table 2. Our mainframe servers (continued)

Server model (Machine type-model)	CPCs CPs	Mode LPs	HSA	Storage: Central Expanded	LCSS	System name, usage Virtual CPs (static, managed) Initial LPAR weight
--------------------------------------	-------------	-------------	-----	---------------------------------	------	--

Notes:

1. For our z900 server, we applied the IYP version of IOCP 1.1.0, which is available with the fix for APAR OW46633 (PTF UW90695). We also applied the fix for HCD APAR OW43131 (PTFs UW99341, UW99342, UW99343, UW99344, UW99345) and the fix for HCM APAR IR43534 (PTFs UR90329 and UR90330).
2. For our z800 server, we applied the fix for IOCP APAR OW52993. We also applied the fix for HCD APAR OW51339.
3. Since z/OS.e is engine licensed, customers must define the MSU capacity of a z/OS.e LP to be on an engine boundary. To do this, IBM recommends using the **Defined capacity** field in the activation profile on the z800 hardware management console (HMC). You must also send to IBM the Transmit System Availability Data (TSAD) for your z800 server, either by using the IBM Remote Support Facility (RSF) on the z800 or by mailing a diskette or DVD cartridge to IBM. For details, see *z/OS and z/OS.e Planning for Installation*, GA22-7504, and *z800 Software Pricing Configuration Technical Paper*, GM13-0121, available from the zSeries Library at www.ibm.com/servers/eserver/zseries/library/.
4. We added several “dummy” logical partitions on our z990 server—LPs that are defined but not activated—in order to force the number of LPs to be greater than 15. Currently, you can define up to 30 LPs on the z990.
5. On the z990 and z890 support elements (SE), you no longer specify an HSA expansion percentage in the activation profile. Instead, the HSA size is now calculated from IOCP MAXDEV value.

Coupling facility details

Table 3 provides information about the coupling facilities in our sysplex. Figure 2 on page 11 further illustrates the coupling facility channel distribution as described in Table 3.

Table 3. Our coupling facilities

Coupling facility name	Model CPCs and CPs CFLEVEL (CFCC level) Controlled by	Storage: Central Expanded	Channel distribution
CF1	zSeries 890 Model A04 (2086-A04) stand-alone coupling facility 1 CPC with 4 CPs CFLEVEL=14 (CFCC Release 14.00, Service Level 00.17) Controlled by the HMC	6G	<p>17 TYPE=CFP channels. These are peer-mode intersystem coupling (ISC) channels².</p> <p>There are 17 corresponding TYPE=CFP channels on the following systems ("shared" indicates that the systems share that number of channels using MIF):</p> <ul style="list-style-type: none"> JG0/JH0: 4 shared J80: 3 JA0/JB0/JC0/JE0/Z2/Z3 and CF2³: 4 shared JF0/J90/TPN/Z0/Z1 and CF3³: 6 shared <hr/> <p>2 TYPE=CBP channels. These are peer-mode integrated cluster bus (ICB) channels².</p> <p>There are 2 corresponding TYPE=CBP channels on the following systems:</p> <ul style="list-style-type: none"> JG0/JH0: 1 JA0/JB0/JC0/JE0/Z2/Z3 and CF2: 1 shared
CF2	Coupling facility LP on a zSeries 990 Model 325 (2084-325) 3 dedicated ICF CPs CFLEVEL=14 (CFCC Release 14.00, Service Level 00.17) Controlled by the HMC	6G	<p>15 TYPE=CFP channels. These are peer-mode ISC channels^{2, 4}.</p> <p>There are 15 corresponding TYPE=CFP channels on the following systems:</p> <ul style="list-style-type: none"> JG0/JH0: 2 shared J80: 3 JF0/J90/TPN/Z0/Z1 and CF3: 6 shared CF1: 4 <hr/> <p>7 TYPE=CBP channels. These are peer-mode ICB channels^{2, 4}.</p> <p>There are 7 corresponding TYPE=CBP channels on the following systems:</p> <ul style="list-style-type: none"> JG0/JH0: 1 shared JF0/J90/TPN/Z0/Z1 and CF3: 5 shared CF1: 1 <hr/> <p>8 TYPE=ICP channels. These are peer-mode internal coupling (IC) channels^{1,2, 4}.</p> <p>There are 8 corresponding TYPE=ICP channels on the following systems:</p> <ul style="list-style-type: none"> JA0/JB0/JC0/JE0/Z2/Z3: 2 shared, spanning LCSS 0 and 2 JB0/JC0/Z2: 4 shared (in LCSS 0) JA0/JE0/Z3: 2 shared (in LCSS 2)

Parallel Sysplex environment

Table 3. Our coupling facilities (continued)

Coupling facility name	Model CPCs and CPs CFLEVEL (CFCC level) Controlled by	Storage: Central Expanded	Channel distribution
CF3	Coupling facility LP on a zSeries 900 Model 212 (2064-212)	6G	16 TYPE=CFP channels ^{2, 3} .
	4 dedicated ICF CPs		There are 16 corresponding TYPE=CFP channels on the following systems:
	CFLEVEL=13 (CFCC Release 13.00, Service Level 04.08)		<ul style="list-style-type: none"> • JG0/JH0: 3 shared • JB0/JC0/Z2: 1 shared • JA0/JE0/Z3 and CF2: 5 shared • JA0/JB0/JC0/JE0/Z2/Z3 and CF2: 1 shared • CF1: 6
	Controlled by the HMC		1 TYPE=CBR channels.
			There is 1 corresponding TYPE=CBS channel shared by systems JG0/JH0.
		9 TYPE=CBP channels ^{2, 3} .	
		There are 9 corresponding TYPE=CBP channels on the following systems:	
		<ul style="list-style-type: none"> • J80: 2 shared • JB0/JC0/Z2: 2 shared • JA0/JE0/Z3 and CF2: 4 shared • JA0/JB0/JC0/JE0/Z2/Z3 and CF2: 1 shared 	
		2 TYPE=ICP channels ^{1, 2, 3} .	
		There are 2 corresponding TYPE=ICP channels shared by systems JF0/J90/TPN/Z0/Z1.	

Notes:

1. Our servers that contain internal coupling facilities (CF2 and CF3) also have internal coupling (IC) channels. IC channels are logical connections between a coupling facility LP and the z/OS LPs on the same CPC. IC channels require no channel or cabling hardware (although CHPID numbers must still be defined in the IOCDs). Because they utilize the system bus, IC channels offer improved coupling performance over intersystem coupling (ISC, channel types CFS, CFR, and CFP) channels and integrated cluster bus (ICB, channel types CBS, CBR, and CBP) channels.
2. On the z800, z890, z900, and z990 servers, you can define coupling facility channels as peer channels on both sides of a coupling facility connection. A peer channel contains both sender and receiver functions; however, it is not necessary for both sides use both functions. You can define ISC, ICB, and IC channels in peer mode as channel types CFP, CBP, and ICP, respectively. You can only use peer mode between coupling facility LPs and z/OS LPs that reside on z800, z890, z900, and z990 servers. See *z/OS HCD Planning* for more information.
3. On our z900 server that contains CF3, all of the peer-mode channels are shared by all of the LPs (z/OS LPs and the coupling facility LP) on the CPC. These paths support the XCF communications both between the z/OS LPs and the other coupling facilities in the sysplex, as well as the system-managed coupling facility structure duplexing between the coupling facility LP and the other coupling facilities in the sysplex.
4. On our z990 server that contains CF2 has two logical channel subsystems (LCSSs). Coupling facility channels that are defined within a single LCSS can only be shared by the LPs (both z/OS images and coupling facility images) in that LCSS. Coupling facility channels that span multiple LCSSs can be shared by the LPs (both z/OS images and coupling facility images) in those LCSSs.

Figure 2 on page 11 illustrates our coupling facility channel configuration.

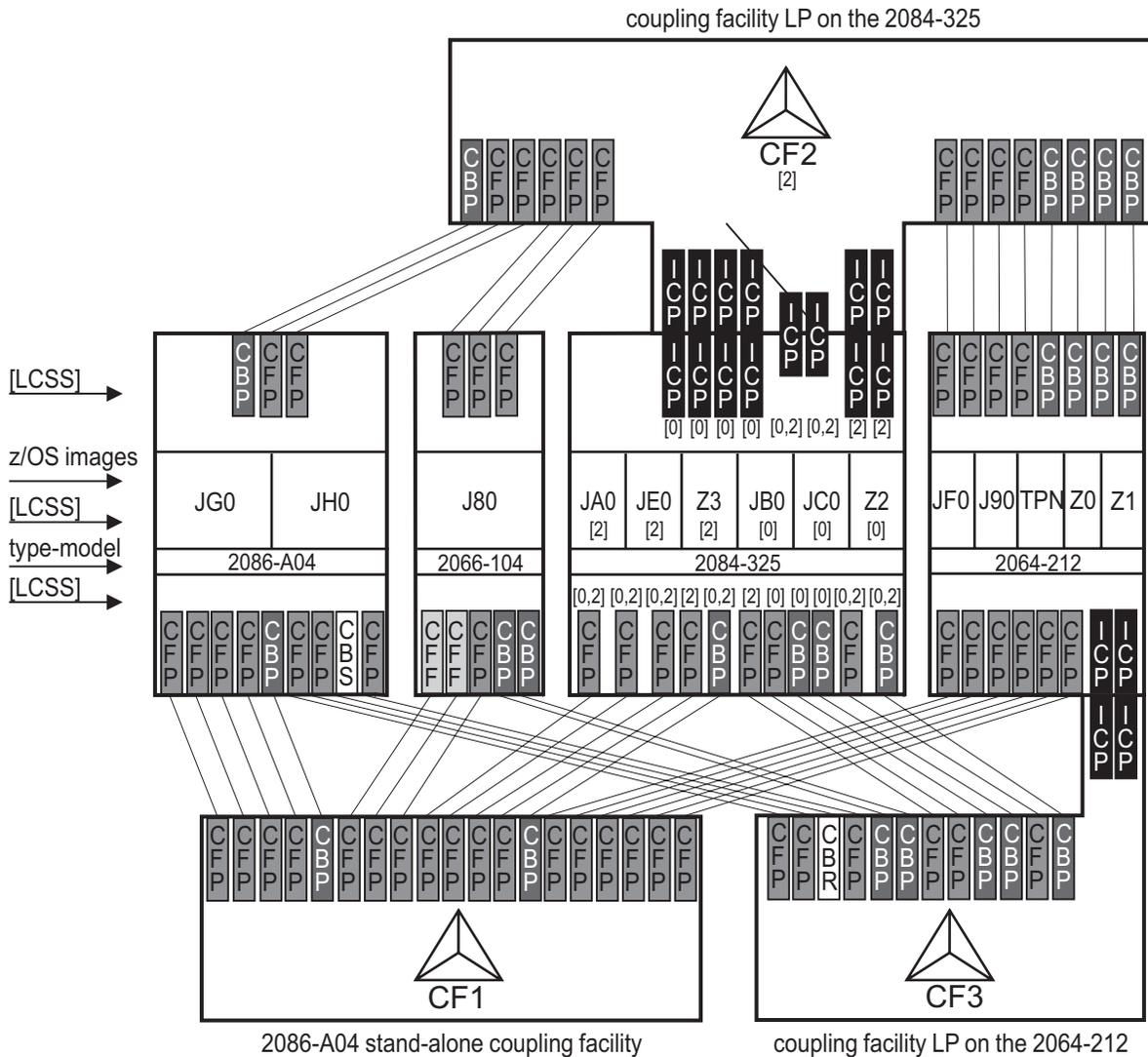


Figure 2. Our coupling facility channel configuration

CFCC Dispatcher Rewrite testing: CFCC Level 14 was installed on the z/990 and the z/890. The major change in CFLevel 14 was the Dispatcher Rewrite. The Dispatcher Rewrite provides the following enhancements:

- It improves efficiency of dispatching CF requests:
 - Incoming and suspended requests are processed in the order they were issued.
 - Ready work is put on a separate queue (rather than remaining on the suspend queue).
 - Latches are processed in the order they were issued.
 - Dispatching and completion of work in progress is favored over executing incoming work.
- It refines the calculation of CF utilization to eliminate time spent searching for work (as opposed to executing work)

Parallel Sysplex environment

One of the things we tested was the migration of some of the CFs to the new level and a mixture of different of CFLEVELs. Table 4 shows some of the different configurations we tested.

Table 4. CFs and CFLEVELS.

CF name	End of Duplex testing	Initial installation	Final installation
CF1	9672 - R06	9672 - R06	2086 - A04
	CFLEVEL 11	CFLEVEL 11	CFLEVEL 14
	7 CPs	7 CPs	4 CPs
CF2	2084	2084	2084
	CFLEVEL 13	CFLEVEL 14	CFLEVEL 14
	3 ICFs	3 ICFs	3 ICFs
CF3	2064	2064	2064
	CFLEVEL 13	CFLEVEL 13	CFLEVEL 13
	4 ICFs	4 ICFs	4 ICFs

As part of our migration to the new CFLevels, we downloaded and ran the recommended CF sizer program, found at:

<http://www.ibm.com/servers/eserver/zseries/cfsizer/altsizer.html>

The program ran against all structures in our CFRM policy. We strongly recommend that you visit this website if you are migrating to a higher level of CFCC. We paid particular attention to the structures running on CFCC level 14 coupling facilities, although in our case, we didn't have to increase any structure sizes.

We ran several stress tests in the various CFCC environments and compared the coupling facility performance data against the baseline we had previously taken to make sure we saw no degradation in performance.

The Duplexing enhancements in CFCC level 14 can only be realized if both copies of a duplexed structure reside on CFs with CFLevel 14 installed. In the final configuration, when we had CFLevel 14 on both CF1 and CF2, we observed a noticeable improvement in our CF to CF service times. We also noted a substantial improvement in the response times of duplexed structures when we replaced the 9672-R06 with the 2086-A04.

Other sysplex hardware details

Table 5 highlights information about the other hardware components in our sysplex:

Table 5. 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.

Table 5. Other sysplex hardware configuration details (continued)

Hardware element	Model or type	Additional information
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 <i>S/390 Parallel Sysplex Performance</i> 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 <i>FICON Native Implementation and Reference Guide</i> , 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 <i>FICON CTC Implementation</i> available on the IBM Redbooks™ Web site.
DASD	Enterprise Storage Server® (ESS, 2105-F20, 800, 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.
Automated tape library (ATL)	3495 Model L40 with 16 additional 3490E tape drives and 12 3590 tape drives	All tape drives are accessible from all systems.
Virtual Tape Server (VTS)	3494 Model L10 with 32 virtual 3490E tape drives and 12 ESCON- and FICON-attached 3590 tape drives	All tape drives are accessible from all systems.

Exploiting 64k cylinder logical volumes

To fulfill our customers requirements for more space, more UCBs and our desire to simplify system administration with fewer devices, we implemented 3390 LVS (large volume support, greater than 32K cylinders volumes) on our IBM DS6000 and IBM DS8000. The first thing we did was to ensure that our ESS licensed internal code (LIC) and our z/OS system software supported this enhancement.

Parallel Sysplex environment

Our large volumes are SMS managed and were initialized with larger VTOCs, as follows:

```
//ICKDSF      EXEC PGM=ICKDSF
//SYSPRINT   DD SYSOUT=*
//SYSABEND   DD SYSOUT=*
//SYSIN      DD *
      INIT UNITADDRESS(6D00) DEVICETYPE(3390) PURGE NORECLAIM NOCHECK SG -
      NOVERIFY VOLID(LVS001) VTOC(2,0,120) INDEX(1,0,15)
```

We exploit dynamic Parallel Access Volume (PAV) and on average have a 12:1 ratio (12 alias' to 1 base).

We use DFSMSHsm to manage space and availability for the data on these volumes, and on all our DASD.

Applications in the zSeries Integration Test environment are exploiting these larger volumes. The applications include: HFS, zFS, DB2, CICS®, IMS, JES, HSM ML1, SVC dumps as well as stand-alone dumps (SAD).

Testing greater than 16 CPU Support

With the combination of z/OS V1R6, z990 servers, and the recommended APARs listed below, customers can now define a single z/OS LPAR image with up to 32 CPs. This function provides the customer more flexibility in choosing the way they want to grow:

- Horizontally with Parallel Sysplex, or
- Vertically using the greater than 16 CP support.

This function is released in two phases:

1. The general availability of z/OS V1R6 in combination with the z990 servers, support up to 24 processors.
2. Later in 2005, this function will be extended to support up to 32 processors.

We installed the following APARs before starting our test:

- OA08993
- OA05907
- OA07200
- OA07857
- OA09340
- OA09688

Our testing occurred in two phases:

1. **Phase 1:** One LPAR, dedicated CPs.

On a z990 server, we defined only one z/OS LPAR, with dedicated CPs. On this image, we ran a high stress level of our IMS, CICS, DB2, MQ, OMVS and WebSphere workloads with a constant number of transactions. We started with 16 CPs online, and then varied CPs online in groups of 8, up to 32 total CPs, while maintaining the same level of transactions. We observed the CPU utilization value and found that it scaled as expected.

2. **Phase 2:** Two LPARs, shared CPs

For this phase we defined two z/OS LPARs, each with 32 shared CPs. On one LPAR we ran the high stress IMS, CICS, DB2, MQ, OMVS and WebSphere

workloads. On the other LPAR we ran low priority workloads (batch and OMVS). The initial weights for the two LPARs were set differently according to the workloads (one higher and one lower).

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 at low levels and slowly increased them until the LPAR was using more than 80% of the total CPU resource. This was done in order to see how WLM and Intelligent Resource Manager (IRD) would manage processor resources. When the high stress LPAR reached 80% CPU utilization, we observed that processors were taken away from the low priority LPAR and that the weights of both LPARs partitions were adjusted accordingly. The number of processors on the high stress LPAR remained at 32 as expected (the maximum allowed).

Here are examples of the RMF™ Monitor III screens that show the number of processors:

```
HARDCOPY      RMF V1R5   CPC Capacity                               Line 1
Command ==>
0Samples: 120      System: J80   Date: 03/11/05   Time: 12.12.00   Range: 120
0Partition:  J80      2084 Model 332
CPC Capacity:    1365   Weight % of Max: ****      4h MSU Average:   163
Image Capacity: 1365   WLM Capping %:  ****      4h MSU Maximum:  234
0Partition --- MSU --- Cap Proc   Logical Util % - Physical Util % -
                Def  Act  Def  Num   Effect  Total  LPAR  Effect  Total
0*CP
J80              0  330 NO   32.0    23.8  24.1   0.3   23.8  24.1
Z1              0   24 NO   32.0     1.7   1.8   0.0    1.7   1.8
PHYSICAL
                0.6                0.6
```

CPU Utilization reached 80% at J80:

```
Time gap from 03/11/05 12.27.00 to 03/11/05 12.27.40.
Samples: 60      System: J80   Date: 03/11/05   Time: 12.26.00   Range: 60
Partition:  J80      2084 Model 332
CPC Capacity:    1365   Weight % of Max: ****      4h MSU Average:   192
Image Capacity: 1365   WLM Capping %:  ****      4h MSU Maximum: 1009
Partition --- MSU --- Cap Proc   Logical Util % - Physical Util % -
                Def  Act  Def  Num   Effect  Total  LPAR  Effect  Total
*CP
J80              0 1139 NO   31.0    85.5  86.1   0.5   82.9  83.4
Z1              0  159 NO   23.0    16.2  16.2   0.0   11.6  11.7
PHYSICAL
                0.3                0.3
```

On/Off Capacity On Demand Testing

As part of IBM's On Demand strategy we tested On/Off Capacity On Demand with the following scenarios:

- z800 Concurrent upgrade testing
- z890 Capacity On Demand testing

z800 Concurrent upgrade testing: We converted our 2066 processor from a model 004 to a model A02. This model conversion was disruptive as you can not concurrently downgrade a z800 processor. At the time of this test we only had one LPAR (J80) on our z800. With all our standard workloads running we concurrently upgraded from a model A02 to a model 002, then to a model 003 and finally back to a model 004. We observed no disruption to our workloads. We did observe an expected decrease in CP utilization while at model types A02, 002, and 003. See "On/Off Capacity on Demand Testing" on page 310 for our RMF screen shots.

z890 Capacity on Demand Testing: With all our normal workloads running, we concurrently downgraded our 2086 processor from a model 370 to a model 360. At the time of this test we had two LPARs on our 2086(JG0,JH0). Each LPAR had two

On/Off Capacity OLTP workloads

general purpose processors and one zAAP processor configured. This downgrade decreased our MSU value from 158 to 91. This drop in MSU capacity only affected our general purpose processors, not our zAAP processors. We ran our 2086 as a model 360 for about two months until we concurrently upgraded our 2086 back to a model 370. During the duration of this test we had no disruptions or problems related to the concurrent downgrade or upgrade.

Our Parallel Sysplex software configuration

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

- CICS Transaction Server (CICS TS) V2R2
- IMS V8.1 (and its associated IRLM)
- DB2 UDB for z/OS and OS/390 V7 (and its associated IRLM)
- WebSphere for z/OS V4.0.1.
- WebSphere MQ for z/OS V5.3.1
- WebSphere Business Integration Message Broker V5.0

We also run z/OS.e in one partition on our z890 server. z/OS.e supports next-generation e-business workloads; it does not support traditional workloads, such as CICS and IMS. However, z/OS.e uses the same code base as z/OS and invokes an operating environment that is identical to z/OS in all aspects of service, management, reporting, and zSeries functionality. See *z/OS.e Overview*, GA22-7869, for more information.

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 and z/OS.e 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 and z/OS.e Planning for Installation* and *z/OS MVS Product Management* for more information about dynamic enablement.

A note about IBM License Manager

In z/OS V1R1, IBM introduced a new base element called IBM License Manager (ILM). IBM has since decided not to deliver the IBM License Manager tool for zSeries. Therefore, when you run z/OS on a z800, z900, or z990 server, you must ensure that the ILMODE parameter in IEASYSxx is set to ILMODE=NONE.

Overview of our software configuration

Figure 3 on page 17 shows a high-level view of our sysplex software configuration.

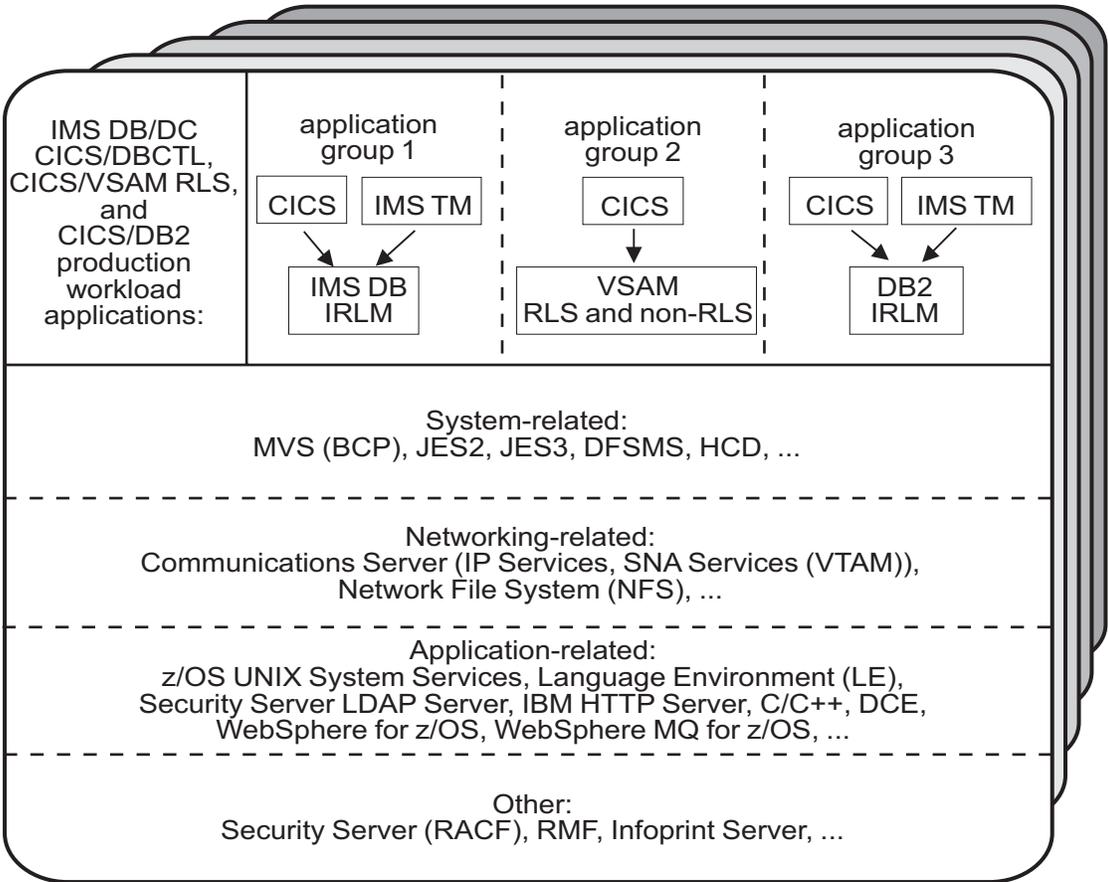


Figure 3. Our sysplex software configuration

We run three separate application groups in one sysplex and each application group spans multiple systems in the sysplex. Table 6 provides an overview of the types of transaction management, data management, and serialization management that each application group uses.

Table 6. Our production OLTP application groups

Application groups	Transaction management	Data management	Serialization management
Group 1	<ul style="list-style-type: none"> • CICS • IMS TM 	IMS DB	IRLM
Group 2	<ul style="list-style-type: none"> • CICS 	VSAM	VSAM record-level sharing (RLS)
Group 3	<ul style="list-style-type: none"> • CICS • IMS TM 	DB2	IRLM

Our December 1995 edition 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 25.

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.

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 are running ICSF, FMID HCR7720, on top of z/OS V1R6. 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:

- Z990:
 - PCI Cryptographic Accelerator (PCICA)
 - PCI X Cryptographic Coprocessor (PCIXCC)
 - Crypto Express2 Coprocessor (CEX2C)
- Z890:
 - PCI X Cryptographic Coprocessor (PCIXCC)
- Z900:
 - Cryptographic Coprocessor Feature (CCF)
 - PCI Cryptographic Accelerator (PCICA)

- Z800:
 - Cryptographic Coprocessor Feature (CCF)

Since our goal is to run a customer-like environment, we have various products running customer-like scenarios using SSL. SSL will in turn use ICSF and any of the Cryptographic Features that we have, as needed. The products that use SSL in our environment are z/OS WebSphere Application Server, FTP, HTTP, and CICS. We also have an ICSF specific workload that runs 16 hours a day, 7 days a week and exercises the cryptographic services available through the ICSF API.

Our networking configuration

For a detailed description of our networking configuration, see Chapter 10, “About our networking and application enablement environment,” on page 123.

Our VTAM configuration

Figure 4 illustrates our current VTAM[®] configuration.

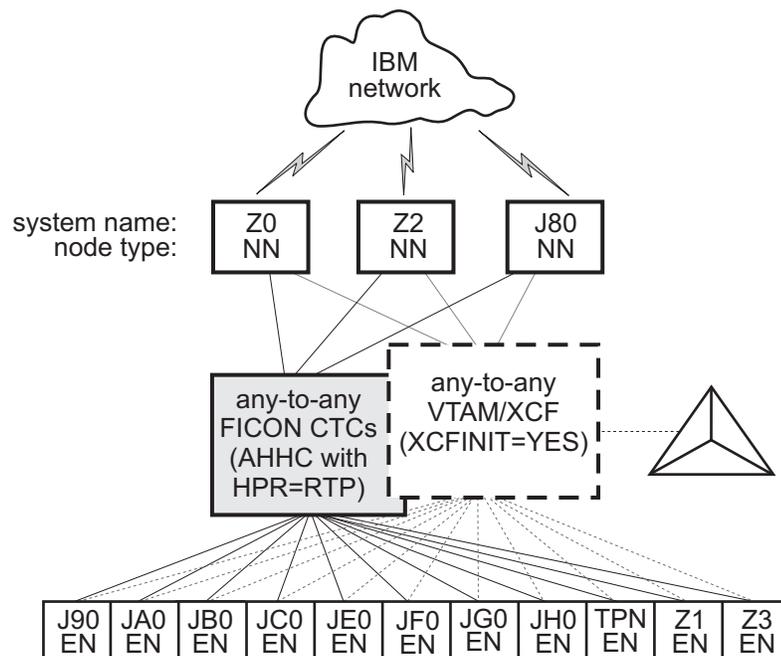


Figure 4. Our VTAM configuration

TPNS runs on our system TPN and routes CICS logons to any of the other systems in the sysplex (except JH0, which runs z/OS.e and does not support CICS).

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.

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

Table 7. Summary of our workloads

Shift	Base system workloads	Application enablement workloads	Networking workloads	Database product workloads
Prime shift	<ul style="list-style-type: none"> Automatic tape switching Batch pipes JES2/JES3 printer simulators 	<ul style="list-style-type: none"> Enterprise Identity Mapping (EIM) IBM HTTP Server LDAP Server z/OS UNIX Shelltest (rlogin/telnet) z/OS UNIX Shelltest (TSO) WebSphere Application Server for z/OS WebSphere MQ WebSphere Business Integration Message Broker MQ batch stress for shared queues MQ-CICS bridge workload MQ connection testing MQ/DB2 bookstore application 	<ul style="list-style-type: none"> AutoWEB FTP workloads MMFACTS for NFS NFSWL Silk Test NFS video stream TCP/IP CICS sockets TN3270 	<ul style="list-style-type: none"> 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	<ul style="list-style-type: none"> Random batch Automatic tape switching JES2/JES3 printer simulators 	<ul style="list-style-type: none"> Enterprise Identity Mapping (EIM) IBM HTTP Server LDAP Server z/OS UNIX Shelltest (rlogin/telnet) z/OS UNIX Shelltest (TSO) WebSphere Application Server for z/OS WebSphere MQ WebSphere Business Integration Message Broker 	<ul style="list-style-type: none"> FTP workloads Silk Test NFS video stream MMFACTS for NFS 	<ul style="list-style-type: none"> 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 edition, 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.

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.

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:

- DB2 through net.data
- MVS data sets
- FastCGI programs
- Counters
- Static html pages
- Protected pages
- SSL transactions

LDAP Server

This workload consists of a script running on a Windows NT® workstation that simulates multiple users running a transaction that drives several different **ldapsearch** commands against the LDAP server on z/OS.

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.

Application enablement workloads

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
- 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

WebSphere MQ workloads

Our WebSphere MQ environment includes one WebSphere MQ for z/OS V5.3.1 queue manager on each system in the sysplex. We have two queue sharing groups: one with three queue managers and another with seven queue managers.

Our workloads test the following WebSphere MQ features:

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

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 via z/OS UNIX shell scripts

The batch-driven 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.

Communications testing: This workload tests our communications channels by kicking off an application that sends messages to a remote queue manager. A trigger monitor program running on the remote system kicks off a separate application that sends the same message back to the originating system. The remote queue manager resides on a mainframe Linux system running WebSphere MQ V5.3.1.

We also run several Web applications to test WebSphere MQ for z/OS. Currently all our applications use the WebSphere Application Server V5.1.

DQM and DQMssl: These workloads test the communication between z/OS queue managers as well as z/OS and Linux queue managers using SSL TCPIP channels and non-SSL APPC 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 a non-Web version, mqcics, that uses shared queues with persistent messages. We defined a separate coupling facility structure for this application.

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 Business Integration Message Broker

Our WebSphere Business Integration Message Broker environment consists of four message brokers: three on test systems, and one on a production system. All are running at WBIMB 5.0.1 FixPack 04. 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 via z/OS UNIX shell scripts

The Web applications consist of html pages, java servlets, and WBIMB 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.

Application enablement workloads

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 that use WBIMB:

Sniffer: This workload tests basic MQ and WBIMB 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 WBIMB 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 WBIMB:

Retail_TPNS: This workload is another version of Retail_IMS, but rather than being driven by WebSphere Studio Workload Simulator, it is driven by TNPS via 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 via 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

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 SMQ 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 recent 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

Database product workloads

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 exercises 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 edition.

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 edition for details on how we set up this application.

Chapter 2. Migrating to and using z/OS

This chapter describes our experiences with migrating to new releases of the z/OS operating system.

Overview

The following sections describe our most recent migration activities:

- “Migrating to z/OS V1R6”
- “Migrating to z/OS.e V1R6” on page 31
- “Migrating to z/OS V1R5” on page 35
- “Migrating to z/OS.e V1R5” on page 48

We primarily discuss our sysplex-related base operating system experiences. This includes the enablement of significant new functions and, if applicable, performance aspects. Detailed test experiences with major new functions beyond migration appear in subsequent chapters.

We discuss our networking and application-enablement environment and test experiences in Part 2, “Networking and application enablement,” on page 119.

You can read about our migration experiences with earlier releases of z/OS and OS/390 in previous editions of our test report, available on our Web site:

For migration experiences with...	See...
z/OS V1R4	our December 2003 edition
z/OS V1R3	our December 2002 edition
z/OS V1R1 and V1R2	our December 2001 edition

Migrating to z/OS V1R6

This section describes our migration experiences with z/OS V1R6.

z/OS V1R6 base migration experiences

In this section we only describe our experiences with our base migration to z/OS V1R6, without having implemented any new functions. It includes our high-level migration process along with other migration activities and considerations.

Our high-level migration process for z/OS V1R6

The following is an overview of our z/OS V1R6 migration process.

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

Table 8 on page 30 shows the high-level process we followed to migrate the members of our sysplex from z/OS V1R5 to z/OS V1R6.

Table 8. Our high-level migration process for z/OS V1R6

Stage	Description
Updating parmlib for z/OS V1R6	We created SYS1.PETR16.PARMLIB to contain all the parmlib members that changed for z/OS V1R6 and we used our LOADxx member for migrating our systems one at a time. (See “Using concatenated parmlib” on page 31 for more about our use of concatenated parmlib and see our December 1997 edition for an example of how we use LOADxx to migrate individual systems.)
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 <i>z/OS and z/OS.e Planning for Installation</i> and make sure you install the fixes for any APARs that relate to your configuration before you migrate.
IPLing our first z/OS V1R6 image	We brought up z/OS V1R6 on our Z1 test system and ran it there for about one week.
Updating the RACF templates	To test the RACF dynamic template enhancement, we IPLed the first z/OS V1R6 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 <i>z/OS Security Server RACF System Programmer's Guide</i> , SA22-7681 for details about RACF templates.)
IPLing additional z/OS V1R6 images	We continued to bring up additional z/OS V1R6 images across our sysplex, as follows: <ul style="list-style-type: none"> • We brought up z/OS V1R6 on our Z2 and Z3 test systems and ran for a couple of days. • Next, we migrated one production system, JF0, and ran for about a week. • Next, we migrated an additional test system, Z1, and two production systems, JG0 and JH0, and ran for another week. • At this point, we took all of the V1R6 images back down to V1R5. This is part of our increased focus on migration testing and fallback. We ran for a full day and experienced no fallback issues. • Next, we migrated an additional test system, Z0, and three production systems, TPN, JB0, and JC0, and ran for a couple of days. • Next, we migrated two more production systems, JA0 and JE0, and ran for about a week. • We then migrated the remaining two systems, J80 and J90.

The total time for our migration was approximately a month.

More about our migration activities for z/OS V1R6

This section highlights additional details about some of our migration activities.

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

- z/OS V1R5 and z/OS V1R6
- z/OS V1R5 and z/OS.e V1R6
- z/OS V1R5 JES2 and z/OS V1R6 JES2
- z/OS V1R5 JES3 and z/OS V1R6 JES3

Using concatenated parmlib: We continue to use concatenated parmlib support to add or update parmlib members for z/OS V1R6. Appendix A, “Some of our parmlib members,” on page 305 summarizes the additions and changes we made by parmlib member. Also 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 V1R6 in one place and makes it easier to migrate multiple systems. Rather than change many parmlib members each time we migrate another system to V1R6, we just add the PARMLIB statements at the appropriate places in SYS0.IPLPARM(LOADxx) to allow that system to use SYS1.PETR16.PARMLIB.

Recompiling REXX EXECs for automation: We recompiled our SA OS/390 REXX EXECs when we migrated to z/OS V1R6. We discuss the need to recompile these REXX EXECs in our our December 1997 edition.

Defining greater than 16 CPs per z/OS image

Beginning with z/OS V1.6, you can now define up to 24 processors in a single z/OS image. Note that the limit of 24 processors is the total of general purpose processors and zAAPs. We defined greater than 16 processors for several of the partitions on our z990 server, up to 22 general purpose processors plus 2 zAAPs. Making this change is done, as usual, from the hardware management console (HMC) by updating the CP panel on the image profiles. Enhancements have been made to several z/OS elements (WLM, etc.) in z/OS V1.6 to support this new limit. IBM intends to support up to 32 processors in a single z/OS image in 2005. See “Testing greater than 16 CPU Support” on page 14 for more information.

Migrating to z/OS.e V1R6

This section describes our migration experiences with z/OS.e V1R6.

z/OS.e V1R6 base migration experiences

This section describes our experiences with migrating one system image (JH0) from z/OS.e V1R5 to z/OS.e V1R6. Here we only cover our experiences with our base migration to z/OS.e V1R6, including our high-level migration process and other migration activities and considerations.

Our high-level migration process for z/OS.e V1R6

The following is an overview of our z/OS.e V1R6 migration process.

Before we began: We reviewed the information in *z/OS and z/OS.e Planning for Installation*, GA22-7504, which covers both z/OS V1R6 and z/OS.e V1R6.

Important notice about cloning and software licensing

As discussed in *z/OS and z/OS.e Planning for Installation*, you might find that sharing system libraries or cloning an already-installed z/OS or z/OS.e system is faster and easier than installing z/OS or z/OS.e with an IBM installation package such as ServerPac. Most Parallel Sysplex customers are already aware of the concept of cloning and the benefits it provides.

However, prior to sharing or cloning z/OS or z/OS.e, **you must have a license for each z/OS and z/OS.e operating system that you run.** If you don't have the appropriate license or licenses, you must contact IBM. Any sharing or cloning of z/OS or z/OS.e without the appropriate licenses is not an authorized use of such programs. On a z800 server, if you want to run both z/OS and z/OS.e, z/OS requires the appropriate license for the machine on which it runs and z/OS.e requires a license for the number of engines on which it runs.

For more information about z/OS.e licensing, see *z800 Software Pricing Configuration Technical Paper* at www.ibm.com/servers/eserver/zseries/library/techpapers/pdf/gm130121.pdf.

Table 9 shows the high-level process we followed to migrate our z/OS.e V1R5 system to z/OS.e V1R6.

Table 9. Our high-level migration process for z/OS.e V1R6

Stage	Description
Obtaining licenses for z/OS.e	You need a license for the appropriate number of engines on the z800 or z890 server on which you intend to run z/OS.e (and, you would also need a license to run z/OS on the z800 or z890, if you intend to install it there). We use an internal process to do this; however, you must use the official process stated in <i>z800 Software Pricing Configuration Technical Paper</i> .
Updating the z800 or z890 LPAR name	z/OS.e must run in LPAR mode and the LPAR name must be of the form Z0SExxxx, where xxxx is up to 4 user-specified alphanumeric characters. The name of the LPAR in which we run z/OS.e is Z0SEJH0. (We used HCD to set this when we first installed z/OS.e V1R3.)
Updating parmlib for z/OS.e V1R6	z/OS.e requires the LICENSE=Z/0SE statement in the IEASYSxx parmlib member. We used the same SYS1.PETR16.PARMLIB data set that we created for z/OS V1R6. We then have separate IEASYSxx and IFAPRDxx members in SYS1.PARMLIB that we tailored specifically for z/OS.e. See "Updating system data sets for z/OS.e" on page 33 for details.
Updating our LOADxx member	During our initial installation of z/OS.e V1R3, we updated the entry for our system JH0 in our LOADxx member in SYS0.IPLPARM to point to our new IEASYS02 parmlib member and to reflect the new LPAR name. Therefore, we did not need to change it for V1R6.

Table 9. Our high-level migration process for z/OS.e V1R6 (continued)

Stage	Description
Updating our IEASYMPT member	During our initial installation of z/OS.e V1R3, we updated the entry for our system JH0 in our IEASYMPT member in SYS1.PETR13.PARMLIB to point to our new IFAPRDxx parmlib member and to reflect the new LPAR name. Therefore, when we created our new SYS1.PETR16.PARMLIB, we carried the change along for V1R6.
IPLing the z/OS.e V1R6 image	We brought up z/OS.e V1R6 on our JH0 production system.

More about our migration activities for z/OS.e V1R6

This section highlights additional details about some of our migration activities.

About our z890 LPAR environment: z/OS.e must run in LPAR mode on a zSeries 800 or 890 mainframe server; it cannot run in basic mode. In addition, the name of the LPAR in which z/OS.e runs must be of the form ZOSExxxx, where xxxx is up to four user-specified alphanumeric characters. The name of our z/OS.e z890 LPAR is ZOSEJH0.

Note: You can only run z/OS.e in a partition named ZOSExxxx. You cannot IPL a z/OS system in a partition named ZOSExxxx.

We currently run z/OS.e (JH0) in a mixed LPAR environment alongside LPARs running z/OS (JG0) on the same z890 server.

Note: Don't let the fact that z/OS.e only runs on a z800 or z890 server confuse you. These are fully functional zSeries servers and, in addition to z/OS.e, they support all of the same zSeries operating systems as a z900 or z990 server.

Updating system data sets for z/OS.e: We continue to use concatenated parmlib support to add or update parmlib members for z/OS.e V1R6. We use the same SYS1.PETR16.PARMLIB data set as we do for our z/OS V1R6 systems.

Below are examples of our parmlib customizations to accommodate z/OS.e V1R6. Appendix A, "Some of our parmlib members," on page 305 summarizes the changes we made by parmlib member.

Example: We have a separate IEASYSxx member, IEASYS02, which specifies the LICENSE=Z/0SE statement that z/OS.e requires.

The entry for our z/OS.e system (JH0) in our LOADxx member in SYS0.IPLPARM points to our IEASYS02 parmlib member and specifies the name of our z/OS.e LPAR, as follows:

```

:
:
HWNAME      z800name
LPARNAME    ZOSEJH0
PARMLIB     SYS1.PETR16.PARMLIB

```

```
SYSPARM 02
:
```

Example: We have a separate IFAPRDxx member, IFAPRD02, which specifies the product ID value 5655-G52 for z/OS.e. There is no change to the product name value for z/OS.e (the product name value remains Z/OS).

Below is an example of one of the entries from our IFAPRD02 member:

```
:
```

```
PRODUCT OWNER('IBM CORP')
        NAME(Z/OS)
        ID( 5655-G52 )
        VERSION(*) RELEASE(*) MOD(*)
        FEATURENAME(Z/OS)
        STATE(ENABLED)
:
```

We also have an entry for our system JH0 in our IEASYMPT member in SYS1.PETR16.PARMLIB to point to our new IFAPRD02 parmlib member and to reflect the z/OS.e LPAR name, as follows:

```
:
```

```
SYSDEF HWNAME(z800name)
        LPARNAME( ZOSEJH0 )
        SYSNAME(JH0)
        SYSCLONE(JH)
:
```

```
:
```

```
        SYMDEF(&PROD= '02' )
:
```

Using current z/OS.e levels of JES2 and LE: As required, we are using the level of JES2 and Language Environment (LE) that comes with z/OS.e V1R6. z/OS.e does not permit the use of a lower level JES2 (or JES3) or LE.

Updating the ARM policy: You must ensure that your automation policies, such as ARM, do not try to use a z/OS.e image to start products that z/OS.e does not support. For example, do not identify a z/OS.e image as a restart target in a Parallel Sysplex that contains a mix of z/OS.e and z/OS images where the z/OS images run IMS, CICS, or DB2 with a requirement for CICS. CICS, IMS, or DB2 that uses CICS cannot restart on a z/OS.e image, but must restart on a z/OS image. If, for example, a CICS region attempts to start on z/OS.e, the region will start but the applications will fail with a U4093 abend.

Back when we installed z/OS.e V1R3, we removed our z/OS.e image, JH0, as a restart target for the unsupported subsystems mentioned above.

Removing z/OS.e from participation in MNPS: In our environment, CICS is the only exploiter of multiple node persistent sessions (MNPS) support. Because CICS cannot run on z/OS.e, there is no reason for the VTAM on z/OS.e to connect to the MNPS structure, ISTMNPS. We removed our z/OS.e image from participating in MNPS by coding the STRMNPS=NONE statement in our VTAM start member, ATCSTRxx, in SYS1.VTAMLST.

Removing z/OS.e from participation in TSO generic resource groups: Since TSO on z/OS.e only allows a maximum of eight concurrent sessions, we removed our z/OS.e image from participating in TSO generic resource groups. You can do this by coding the GNAME=NONE parameter—either in a separate TSOKEYxx member in parmlib or on the START command that starts the terminal control address space (TCAS).

In our case, we use a single TSOKEYxx member that has a symbolic value for the GNAME parameter. We then set that symbol to NONE for our JH0 image in our IEASYMPT member.

Other experiences with z/OS.e V1R6

Our testing of z/OS.e V1R6 included the following workloads or scenarios:

- z/OS UNIX System Services
- DB2 UDB
- IBM HTTP Server in scalable server mode
- WebSphere Application Server for z/OS
- CICS Transaction Gateway (CTG) to access CICS regions running in z/OS images on the same CPC and other CPCs
- DB2 access from Linux guests under z/VM on the same CPC
- our Bookstore application transactions

Migrating to z/OS V1R5

This section describes our migration experiences with z/OS V1R5.

z/OS V1R5 base migration experiences

In this section we only describe our experiences with our base migration to z/OS V1R5, without having implemented any new functions. It includes our high-level migration process along with other migration activities and considerations.

Our high-level migration process for z/OS V1R5

The following is an overview of our z/OS V1R5 migration process.

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

Table 10 shows the high-level process we followed to migrate the members of our sysplex from z/OS V1R4 to z/OS V1R5.

Table 10. Our high-level migration process for z/OS V1R5

Stage	Description
Updating parmlib for z/OS V1R5	We created SYS1.PETR15.PARMLIB to contain all the parmlib members that changed for z/OS V1R5 and we used our LOADxx member for migrating our systems one at a time. (See “Using concatenated parmlib” on page 36 for more about our use of concatenated parmlib and see our December 1997 edition for an example of how we use LOADxx to migrate individual systems.)
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 <i>z/OS and z/OS.e Planning for Installation</i> and make sure you install the fixes for any APARs that relate to your configuration before you migrate.
IPLing our first z/OS V1R5 image	We brought up z/OS V1R5 on our Z1 test system and ran it there for about one week.

Table 10. Our high-level migration process for z/OS V1R5 (continued)

Stage	Description
Updating the RACF templates	<p>To test the RACF dynamic template enhancement, we IPLed the first z/OS V1R5 image without first running the IRRMIN00 utility with PARM=UPDATE. As expected, the following message appeared:</p> <pre>ICH579E RACF TEMPLATES ON DATABASE ARE DOWNLEVEL</pre> <p>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 <i>z/OS Security Server RACF System Programmer's Guide</i>, SA22-7681 for details about RACF templates.)</p>
IPLing additional z/OS V1R5 images	<p>We continued to bring up additional z/OS V1R5 images across our sysplex, as follows:</p> <ul style="list-style-type: none"> • We brought up z/OS V1R5 on our Z2 test system and ran for a couple of days. • Next, we migrated three production systems, JA0, JG0, and JH0 (z/OS.e), to z/OS V1R5 and ran a couple of days. • At this point, we took all of the V1R5 images back down to V1R4 feature 2. This is part of our increased focus on migration testing and fallback. We ran for a full day and experienced no fallback issues. • We migrated systems Z1, Z2, JA0, JG0, and JH0 back to V1R5, along with production systems JF0, TPN, and Z0. We ran this way for about one week. • We then migrated the remaining six systems, JB0, JC0, JE0, J80, J90, and Z3, to z/OS V1R5.

The total time for our migration was approximately three weeks.

More about our migration activities for z/OS V1R5

This section highlights additional details about some of our migration activities.

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

- z/OS V1R4 and z/OS V1R5
- z/OS V1R4 and z/OS.e V1R5
- z/OS V1R4 JES2 and z/OS V1R5 JES2
- z/OS V1R4 JES3 and z/OS V1R5 JES3

Using concatenated parmlib: We continue to use concatenated parmlib support to add or update parmlib members for z/OS V1R5. Appendix A, "Some of our parmlib members," on page 305 summarizes the additions and changes we made by parmlib member. Also 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 V1R5 in one place and makes it easier to migrate multiple systems. Rather than change many parmlib members each time we migrate another system to V1R5, we just add the PARMLIB statements at the appropriate places in SYS0.IPLPARM(LOADxx) to allow that system to use SYS1.PETR15.PARMLIB.

Recompiling REXX EXECs for automation: We recompiled our SA OS/390 REXX EXECs when we migrated to z/OS V1R5. We discuss the need to recompile these REXX EXECs in our our December 1997 edition.

Installing related service: We needed to install the fix for IMS APAR PQ81018 to fix a problem that prevented us from running transactions that accessed Fast Path databases (DEDBs). The APAR resolves an incompatibility between IMS V8 DEDB processing and z/OS V1R5 which results in an S0C4 abend in module DBFMMIT0 during DEDB area open processing.

Using DFSMS enhanced data integrity for sequential data sets

Data integrity for shared, sequential data sets (DISP=SHR) has been enhanced in z/OS DFSMS V1R5. When you activate the enhanced data integrity function, you can prevent accidental data loss by setting the sharing specifications for sequential data sets that are opened for output or update. When the function is active, DFSMS reports a violation if a program attempts to open a sequential data set for writing when the data set is already open for writing.

Parameters to control the enhanced data integrity function reside in a new, customer-created parmlib member, IFGPSEDI. In this member, you can specify whether the enhanced data integrity function is to operate in WARN mode or ENFORCE mode when a violation occurs. You can also specify a list of data sets to exclude from enhanced data integrity processing.

Table 11 summarizes how the enhanced data integrity function behaves in each operating mode.

Table 11. Summary of the modes of operation for the DFSMS enhanced data integrity function for sequential data sets

When a violation occurs and the enhanced data integrity mode is...	The OPEN request proceeds as follows:
MODE(WARN)	The program issues a warning message (IEC984I or IEC985I) when an application attempts to open for input or output a shared, sequential data set that is already open for output, even if the data set is in the exclude list. You can use this information to analyze the violations and determine which data sets to exclude from data integrity processing or which applications to modify.
MODE(ENFORCE)	The program abends when an application attempts to open for output a shared, sequential data set that is already open for output, unless the data set is in the exclude list. No warning messages are issued.
MODE(DISABLE) or the IFGPSEDI member does not exist	The enhanced data integrity function is inactive. Application processing continues as before.

We are currently running with enhanced data integrity in WARN mode. We have entries in the exclude list for the DFSMSshm journal, DFSMSrmm journal, IMS, and JES3 checkpoint data sets.

Example: The following is an example of the contents of our SYS1.PARMLIB(IFGPSEDI) member:

```
MODE(WARN)                /* PS ENHANCED DATA INTEGRITY - WARNING MODE
DSN(D10.PET.DFRMM.JRN)    /* EXCLUDE LIST
DSN(HSM.PET.JRNL)
DSN(DBS%.IMS%.*)
```

```
DSN(DBS%.DSW4.**)  
DSN(SYS1.PET.JES3*)  
DSN(SYS1.CDS1%)  
DSN(SYS1.*.CDS1%)
```

For more information about setting up this new function, see *z/OS Migration*.

Using the new XCF REALLOCATE process

After a coupling facility structure is allocated, the system makes no attempt to optimize the placement of that structure with respect to the installation's wishes (as expressed in the active CFRM policy). Yet, there are many reasons why, over time, structures that were initially allocated in their installation-desired coupling facilities can and do move to other sub-optimal (from the installation's standpoint) coupling facilities. The use of existing commands to restore structures to their desired locations is a complex and often error-prone process, especially in cases where structures support user-managed or system-managed duplexing and the installation has three or more coupling facilities.

APAR OA03481 addresses this problem by providing a new XCF REALLOCATE process which is controlled by enhanced SETXCF commands. The lowest release that supports this enhancement is z/OS V1R4. In order to exploit the enhancement, the PTF for OA03481 must be installed prior to IPLing the system into the sysplex. All systems in the sysplex must support the REALLOCATE process in order to exploit this enhancement.

The REALLOCATE process will **not** be started when XCF discovers an active system in the sysplex which does not have support for the REALLOCATE process installed. If a REALLOCATE process is in progress and a system without support for the REALLOCATE process joins the sysplex, processing will terminate immediately, allowing the structure that is currently in rebuild processing to complete the current process but no additional processing (for example, reduplexing of the structure) will occur.

The intended use of the REALLOCATE process is to simplify coupling facility-related procedures, such as the following:

- To move structures out of a coupling facility following a CFRM policy change that deletes or changes that coupling facility (for example, in preparation for a coupling facility upgrade)
- To move structures back into a coupling facility following a CFRM policy change that adds or restores the coupling facility (for example, following a coupling facility upgrade or addition)
- To clean up pending CFRM policy changes that may have accumulated for whatever reason, even in the absence of a need to actually relocate any structures
- To clean up simplex or duplexed structures that were allocated in or moved into the "wrong" coupling facilities for whatever reason (for example, the "right" coupling facility was inaccessible at the time of allocation)
- To clean up duplexed structures that have their primary and secondary instances "reversed" due to a prior condition that resulted in stopping duplexing with KEEP=NEW and the structure reduplexed

The REALLOCATE process uses existing XCF structure allocation algorithms to recognize the need to relocate structure instances by comparing each structure's current location with the location selected by allocation criteria using either the active or pending CFRM policy. When the locations differ or a policy change is

pending, the REALLOCATE process uses the structure rebuild process to make the necessary adjustments. The following types of structure rebuild processes are supported:

- user-managed rebuild
- user-managed duplexing rebuild
- system-managed rebuild
- system-managed duplexing rebuild

The following SETXCF operator commands have been enhanced to support the REALLOCATE process:

- **SETXCF START,REALLOCATE**

This command starts the REALLOCATE process. Each allocated structure (simplex or duplexed) is evaluated. Once selected as the target of the REALLOCATE process, predetermined steps are used to relocate the instances or activate a pending policy change. Messages IXC543I, IXC544I, IXC545I, and IXC546I are issued for tracking the REALLOCATE process. Existing messages IXC52nI and IXC57nI are issued for structure rebuild processing. Message IXC574I is written to the log with the evaluation information for a structure.

- **SETXCF STOP,REALLOCATE**

This command stops the REALLOCATE process after processing of the current target structure completes.

- **SETXCF STOP,REALLOCATE,FORCE**

This commands immediately stops the REALLOCATE process. The structure rebuild processing for the target structure is allowed to complete but, for a duplexed structure, the steps to relocate or reduplex the structure may not be done. The FORCE option should be used when structure rebuild processing for the target structure is not making progress.

The remainder of this topic shows several examples of using the enhanced SETXCF commands to control the REALLOCATE process and the resulting messages.

Example: The following is an example of issuing the SETXCF START,REALLOCATE command from a system with APAR OA03481 installed that is in a sysplex with at least one down-level system (that is, a system at a release level lower than z/OS V1R4 or that does not have APAR OA03481 installed).

```
12:59:14.93 MARTHA 00000200 SETXCF START,REALLOCATE
12:59:14.97 MARTHA 01000000 IXC543I THE REQUESTED START,REALLOCATE WAS REJECTED
395 01000000 AT LEAST ONE SYSTEM DOES NOT SUPPORT THE REALLOCATE PROCESS
```

Example: The following is an example of the response from issuing the SETXCF START,REALLOCATE command from a system at a release level lower than z/OS V1R4 or that does not have APAR OA03481 installed.

```
13:19:24.75 MARTHA 01000000 SETXCF SYNTAX ERROR, COULD NOT RECOGNIZE: REALLOCATE.
839 01000000 ONE OF THE FOLLOWING WAS EXPECTED:
839 01000000 PATHIN PATHOUT CLASSDEF POLICY
839 01000000 REBUILD ALTER
```

Example: When you issue the SETXCF START,REALLOCATE command in a sysplex in which all systems have APAR OA03481 installed, REALLOCATE processing begins. The following log excerpts show the results for the case where changes were made to the preference list in the CFRM policy and the REALLOCATE process was used to process and complete the pending policy changes. Commentary has been inserted as appropriate to describe the progress of the REALLOCATE processing.

SETXCF START,REALLOCATE
 IXC543I THE REQUESTED START,REALLOCATE WAS ACCEPTED.

(Duplexed structure DSND1G_LOCK1 is evaluated.)

IXC574I EVALUATION INFORMATION FOR REALLOCATE PROCESSING 486
 OF STRUCTURE DSND1G_LOCK1

DUPLEXED STRUCTURE ALLOCATED IN COUPLING FACILITY: CF2 CF3
 ACTIVE POLICY INFORMATION USED.
 CFNAME STATUS/FAILURE REASON

 CF2 PREFERRED CF 1
 INFO110: 00000064 CC007B00 0000000D
 CF3 PREFERRED CF 2
 INFO110: 00000064 CC007B00 0000000D
 CF1 PREFERRED CF ALREADY SELECTED
 INFO110: 00000064 CC007B00 0000000D

(The structure is not selected as a target structure.)

IXC544I REALLOCATE PROCESSING FOR STRUCTURE DSND1G_LOCK1 487
 WAS NOT ATTEMPTED BECAUSE
 STRUCTURE IS ALLOCATED IN PREFERRED CF

IXC574I EVALUATION INFORMATION FOR REALLOCATE PROCESSING 488
 OF STRUCTURE DSND1G_GBP0

DUPLEXED STRUCTURE ALLOCATED IN COUPLING FACILITY: CF3 CF2
 ACTIVE POLICY INFORMATION USED.
 CFNAME STATUS/FAILURE REASON

 CF3 PREFERRED CF 1
 INFO110: 00000064 CC007800 0000000D
 CF2 PREFERRED CF 2
 INFO110: 00000064 CC007800 0000000D
 CF1 PREFERRED CF ALREADY SELECTED
 INFO110: 00000064 CC007800 0000000D

IXC544I REALLOCATE PROCESSING FOR STRUCTURE DSND1G_GBP0 489
 WAS NOT ATTEMPTED BECAUSE
 STRUCTURE IS ALLOCATED IN PREFERRED CF

IXC574I EVALUATION INFORMATION FOR REALLOCATE PROCESSING 490
 OF STRUCTURE DSND1G_GBP1

DUPLEXED STRUCTURE ALLOCATED IN COUPLING FACILITY: CF2 CF3
 ACTIVE POLICY INFORMATION USED.
 CFNAME STATUS/FAILURE REASON

 CF2 PREFERRED CF 1
 INFO110: 0000008C AC007800 0000000D
 CF3 PREFERRED CF 2
 INFO110: 0000008C AC007800 0000000D
 CF1 PREFERRED CF ALREADY SELECTED
 INFO110: 0000008C AC007800 0000000D

IXC544I REALLOCATE PROCESSING FOR STRUCTURE DSND1G_GBP1 491
 WAS NOT ATTEMPTED BECAUSE
 THE STRUCTURE HAS NO ACTIVE CONNECTORS

IXC574I EVALUATION INFORMATION FOR REALLOCATE PROCESSING 492
 OF STRUCTURE DSND1G_GBP2

DUPLEXED STRUCTURE ALLOCATED IN COUPLING FACILITY: CF3 CF2
 ACTIVE POLICY INFORMATION USED.
 CFNAME STATUS/FAILURE REASON

 CF3 PREFERRED CF 1
 INFO110: 00000046 CC007800 0000000D
 CF2 PREFERRED CF 2
 INFO110: 00000046 CC007800 0000000D

```

CF1          PREFERRED CF ALREADY SELECTED
              INFO110: 00000046 CC007800 0000000D
IXC544I REALLOCATE PROCESSING FOR STRUCTURE DSNDB1G_GBP2 493
WAS NOT ATTEMPTED BECAUSE
STRUCTURE IS ALLOCATED IN PREFERRED CF
  :
```

(Additional log output omitted to condense the display.)

```

IXC574I EVALUATION INFORMATION FOR REALLOCATE PROCESSING 509
OF STRUCTURE DSNDB1G_SCA
  DUPLEXED STRUCTURE ALLOCATED IN COUPLING FACILITY: CF2      CF3
  PENDING POLICY INFORMATION USED.
```

(Structure DSNDB1G_SCA is evaluated using the pending policy. The REALLOCATE process will cause the pending policy change to happen.)

CFNAME	STATUS/FAILURE REASON
CF2	PREFERRED CF 1 INFO110: 00000064 CC007B00 0000000D
CF3	PREFERRED CF 2 INFO110: 00000064 CC007B00 0000000D

(Structure DSNDB1G_SCA becomes the target structure.)
(Step 1 — Stop duplexing.)

```

IXC522I SYSTEM-MANAGED DUPLEXING REBUILD FOR STRUCTURE 510
DSNDB1G_SCA IS BEING STOPPED
TO FALL BACK TO THE OLD STRUCTURE DUE TO
REQUEST FROM AN OPERATOR
IXC571I SYSTEM-MANAGED DUPLEXING REBUILD FOR STRUCTURE 511
DSNDB1G_SCA HAS COMPLETED THE DUPLEX ESTABLISHED PHASE
AND IS ENTERING THE QUIESCE FOR STOP PHASE.
TIME: 06/16/2004 09:59:04.236728
AUTO VERSION: BB51F5FD F1E7C385
  :
```

(Additional log output omitted....)

```

IXC571I SYSTEM-MANAGED DUPLEXING REBUILD FOR STRUCTURE 946
DSNDB1G_SCA HAS COMPLETED THE QUIESCE FOR STOP PHASE
AND IS ENTERING THE STOP PHASE.
TIME: 06/16/2004 09:59:06.102207
AUTO VERSION: BB51F5FD F1E7C385
IXC577I SYSTEM-MANAGED DUPLEXING REBUILD HAS 947
BEEN STOPPED FOR STRUCTURE DSNDB1G_SCA
STRUCTURE NOW IN COUPLING FACILITY CF2
PHYSICAL STRUCTURE VERSION: BB47CF06 640AC8B5
LOGICAL STRUCTURE VERSION: BB47CF06 640AC8B5
AUTO VERSION: BB51F5FD F1E7C385
  :
```

(Additional log output omitted....)

```

IXC579I PENDING DEALLOCATION FOR STRUCTURE DSNDB1G_SCA IN 948
      COUPLING FACILITY 002064.IBM.02.00000002A48A
      PARTITION: 04      CPCID: 00
HAS BEEN COMPLETED.
PHYSICAL STRUCTURE VERSION: BB51F601 0CC58370
```

INF0116: 13088068 01 6A00 00000014
:

(Additional log output omitted....)

(Step 2 — Rebuild the structure. As the structure exists in the preferred CF, rebuild is in place to cause the policy change to take effect.)

IXC521I REBUILD FOR STRUCTURE DSNDB1G_SCA 950
HAS BEEN STARTED
IXC526I STRUCTURE DSNDB1G_SCA IS REBUILDING FROM 935
COUPLING FACILITY CF2 TO COUPLING FACILITY CF2.
REBUILD START REASON: OPERATOR INITIATED
INFO108: 00000064 00000064.
IXL014I IXLCONN REBUILD REQUEST FOR STRUCTURE DSNDB1G_SCA 936
WAS SUCCESSFUL. JOBNAME: DB91MSTR ASID: 017D
CONNECTOR NAME: DB2_DB91 CFNAME: CF2
IXL015I REBUILD NEW STRUCTURE ALLOCATION INFORMATION FOR 937
STRUCTURE DSNDB1G_SCA, CONNECTOR NAME DB2_DB91
CFNAME ALLOCATION STATUS/FAILURE REASON

CF2 STRUCTURE ALLOCATED CC007B00
CF3 PREFERRED CF ALREADY SELECTED CC007B00
:

(Additional log output omitted....)

IXC521I REBUILD FOR STRUCTURE DSNDB1G_SCA 140
HAS BEEN COMPLETED
:

(Additional log output omitted....)

SCA STRUCTURE DSNDB1G_SCA REBUILD SUCCESSFUL.
IXC579I PENDING DEALLOCATION FOR STRUCTURE DSNDB1G_SCA IN 141
COUPLING FACILITY 002084.IBM.00.00000001B52A
PARTITION: 23 CPCID: 00
HAS BEEN COMPLETED.
PHYSICAL STRUCTURE VERSION: BB47CF06 640AC8B5
INFO116: 13088068 01 6A00 00000001
TRACE THREAD: 00029DA7.
:

(Additional log output omitted....)

(Step 3 — Re-duplex the structure.)

IXC570I SYSTEM-MANAGED DUPLEXING REBUILD STARTED FOR STRUCTURE 148
DSNDB1G_SCA IN COUPLING FACILITY CF2
PHYSICAL STRUCTURE VERSION: BB60B035 01AF4B24
LOGICAL STRUCTURE VERSION: BB60B035 01AF4B24
START REASON: OPERATOR-INITIATED
AUTO VERSION: BB60B046 8526F7A7
:

(Additional log output omitted....)

(Simplex structure IXCPLEX_PATH4 is evaluated.)

IXC574I EVALUATION INFORMATION FOR REALLOCATE PROCESSING 656
OF STRUCTURE IXCPLEX_PATH4
SIMPLEX STRUCTURE ALLOCATED IN COUPLING FACILITY: CF1
ACTIVE POLICY INFORMATION USED.
CFNAME STATUS/FAILURE REASON

```

CF1      PREFERRED CF 1
                INFO110: 0000008C CC007800 0000000D
CF3      PREFERRED CF ALREADY SELECTED
                INFO110: 0000008C CC007800 0000000D
CF2      PREFERRED CF ALREADY SELECTED
                INFO110: 0000008C CC007800 0000000D

```

(The structure is not selected as a target structure.)

```

IXC544I REALLOCATE PROCESSING FOR STRUCTURE IXCPLEX_PATH4 657
WAS NOT ATTEMPTED BECAUSE
STRUCTURE IS ALLOCATED IN PREFERRED CF
:

```

(Additional log output omitted....)

(Simplex structure IRLMLOCKT is evaluated.)

```

IXC574I EVALUATION INFORMATION FOR REALLOCATE PROCESSING 997
OF STRUCTURE IRLMLOCKT
SIMPLEX STRUCTURE ALLOCATED IN COUPLING FACILITY: CF1
PENDING POLICY INFORMATION USED.

```

(Structure IRLMLOCKT is evaluated using the pending policy. Based on the pending policy change, the structure is not allocated in the most preferred CF; therefore, reallocation is required.)

CFNAME	STATUS/FAILURE REASON
CF3	PREFERRED CF 1 INFO110: 0000008C AC007800 0000000D
CF2	PREFERRED CF ALREADY SELECTED INFO110: 0000008C AC007800 0000000D

(Structure IRLMLOCKT becomes the target structure as it is currently allocated in a less-preferred CF.)
(Start rebuild for the structure.)

```

IXC570I SYSTEM-MANAGED REBUILD STARTED FOR STRUCTURE 998
IRLMLOCKT IN COUPLING FACILITY CF1
PHYSICAL STRUCTURE VERSION: BB3DBE1C C419E1AC
LOGICAL STRUCTURE VERSION: BB3DBE1C C419E1AC
START REASON: OPERATOR-INITIATED
AUTO VERSION: BB60B0A4 24906247

```

```

IXC578I SYSTEM-MANAGED REBUILD SUCCESSFULLY ALLOCATED 999
STRUCTURE IRLMLOCKT.

```

```

OLD COUPLING FACILITY: CF1
OLD PHYSICAL STRUCTURE VERSION: BB3DBE1C C419E1AC
NEW COUPLING FACILITY: CF3
NEW PHYSICAL STRUCTURE VERSION: BB60B0A4 5DDCFD25
LOGICAL STRUCTURE VERSION: BB3DBE1C C419E1AC
AUTO VERSION: BB60B0A4 24906247
IXC574I ALLOCATION INFORMATION FOR SYSTEM-MANAGED REBUILD 000
OF STRUCTURE IRLMLOCKT
AUTO VERSION: BB60B0A4 24906247

```

CFNAME	STATUS/FAILURE REASON
CF3	STRUCTURE ALLOCATED INFO110: 0000008C AE007800 00000000
CF2	PREFERRED CF ALREADY SELECTED INFO110: 0000008C AE007800 0000000D

```

IXC571I SYSTEM-MANAGED REBUILD FOR STRUCTURE 001
IRLMLOCKT HAS COMPLETED THE ALLOCATION PHASE
AND IS ENTERING THE COPY PHASE.

```

```

TIME: 06/16/2004 10:01:17.269945
AUTO VERSION: BB60B0A4 24906247

```

```

IXC572I SYSTEM-MANAGED REBUILD FOR STRUCTURE 610
IRLMLOCKT HAS COMPLETED THE INITIALIZATION

```

SUBPHASE OF THE COPY PHASE AND IS ENTERING THE ATTACH SUBPHASE.
 TIME: 06/16/2004 10:01:17.424153
 AUTO VERSION: BB60B0A4 24906247
 IXC572I SYSTEM-MANAGED REBUILD FOR STRUCTURE 611
 IRLMLOCKT HAS COMPLETED THE ATTACH SUBPHASE OF THE COPY PHASE AND IS ENTERING THE LIST SUBPHASE.
 TIME: 06/16/2004 10:01:17.438925
 AUTO VERSION: BB60B0A4 24906247
 IXC572I SYSTEM-MANAGED REBUILD FOR STRUCTURE 171
 IRLMLOCKT HAS COMPLETED THE LIST SUBPHASE OF THE COPY PHASE AND IS ENTERING THE EXIT SUBPHASE.
 TIME: 06/16/2004 10:01:17.813450
 AUTO VERSION: BB60B0A4 24906247
 :

(Additional log output omitted....)

IXC579I NORMAL DEALLOCATION FOR STRUCTURE IRLMLOCKT IN 612
 COUPLING FACILITY 002086.IBM.02.00000002AE1A
 PARTITION: 01 CPCID: 00
 HAS BEEN COMPLETED.
 PHYSICAL STRUCTURE VERSION: BB3DBE1C C419E1AC
 INF0116: 130AA000 FF FFFF 00000010
 TRACE THREAD: 0009C98B.
 IXC577I SYSTEM-MANAGED REBUILD HAS 613 BEEN COMPLETED FOR STRUCTURE IRLMLOCKT
 STRUCTURE NOW IN COUPLING FACILITY CF3
 PHYSICAL STRUCTURE VERSION: BB3DBE1C C419E1AC
 LOGICAL STRUCTURE VERSION: BB3DBE1C C419E1AC
 AUTO VERSION: BB60B0A4 24906247
 IXC580I SYSTEM-MANAGED REBUILD OF STRUCTURE IRLMLOCKT 614
 AUTO VERSION: BB60B0A4 24906247
 RESULTED IN THE FOLLOWING STRUCTURE ATTRIBUTES:
 PHYSICAL STRUCTURE VERSION: BB3DBE1C C419E1AC
 LOGICAL STRUCTURE VERSION : BB3DBE1C C419E1AC
 CURRENT SIZE : 8192 K
 CURRENT ENTRY COUNT : 11531
 CURRENT ELEMENT COUNT : 0
 CURRENT EMC COUNT : 0
 :

(Additional log output omitted....)

(Summary of REALLOCATE processing. Note that some of the structure processing referenced in the summary was omitted from this example display.)

IXC545I REALLOCATE PROCESSING RESULTED IN THE FOLLOWING: 527
 10 STRUCTURE(S) REALLOCATED - SIMPLEX
 11 STRUCTURE(S) REALLOCATED - DUPLEXED
 0 STRUCTURE(S) POLICY CHANGE MADE - SIMPLEX
 17 STRUCTURE(S) POLICY CHANGE MADE - DUPLEXED
 48 STRUCTURE(S) ALREADY ALLOCATED IN PREFERRED CF - SIMPLEX
 43 STRUCTURE(S) ALREADY ALLOCATED IN PREFERRED CF - DUPLEXED
 5 STRUCTURE(S) NOT PROCESSED
 167 STRUCTURE(S) NOT ALLOCATED
 19 STRUCTURE(S) NOT DEFINED

 320 TOTAL
 0 ERROR(S) ENCOUNTERED DURING PROCESSING

(REALLOCATE processing complete.)


```

000000 STRNAME: CSLRMGR_PROD
000000 STATUS: ALLOCATED
000000 REALLOCATE EVALUATION PENDING
000000 TYPE: SERIALIZED LIST
000000 POLICY INFORMATION:
000000 POLICY SIZE : 32000 K
000000 POLICY INITSIZE: 20000 K
000000 POLICY MINSIZE : 15000 K
000000 FULLTHRESHOLD : 60
000000 ALLOWAUTOALT : YES
000000 REBUILD PERCENT: N/A
000000 DUPLEX : ALLOWED
000000 PREFERENCE LIST: CF2 CF1 CF3
000000 ENFORCEORDER : NO
000000 EXCLUSION LIST IS EMPTY
000000 ACTIVE STRUCTURE
000000 -----
000000 ALLOCATION TIME: 05/17/2004 21:37:02
000000 CFNAME : CF3
000000 COUPLING FACILITY: 002064.IBM.02.00000002A48A
000000 PARTITION: 04 CPCID: 00
000000 ACTUAL SIZE : 20224 K
000000 STORAGE INCREMENT SIZE: 256 K
000000 PHYSICAL VERSION: BB3B9434 0EEDFB40
000000 LOGICAL VERSION: BB3B9434 0EEDFB40
000000 SYSTEM-MANAGED PROCESS LEVEL: 9
000000 XCF GRPNAME : IXCL00E1
000000 DISPOSITION : KEEP
000000 ACCESS TIME : NOLIMIT
000000 MAX CONNECTIONS: 32
000000 # CONNECTIONS : 1
:

```

(Additional log output omitted....)

Example: The following is an example of issuing the DISPLAY XCF,STR command during REALLOCATE processing.

```

IXC359I 13.49.12 DISPLAY XCF 517
THE REALLOCATE PROCESS IS IN PROGRESS.
STRNAME ALLOCATION TIME STATUS
APPCLOG 02/25/2004 17:08:37 ALLOCATED
REALLOCATE EVALUATION PENDING

```

(The REALLOCATE process evaluates structures in a serial manner and has not yet reached this structure.)

```

CICS_USERJRN_001 03/03/2004 13:28:14 ALLOCATED
REALLOCATE EVALUATION PENDING
COUPLE_CKPT1 03/02/2004 20:40:48 DUPLEXING REBUILD NEW STRUCTURE
DUPLEXING REBUILD
METHOD : SYSTEM-MANAGED
REBUILD PHASE: DUPLEX ESTABLISHED
REALLOCATE EVALUATION PENDING
COUPLE_CKPT1 02/24/2004 23:40:51 DUPLEXING REBUILD OLD STRUCTURE
DUPLEXING REBUILD
NOT ALLOCATED
COUPLE_CKPT2 -- --
CQS_FF_LOGSTR 03/03/2004 08:26:14 DUPLEXING REBUILD NEW STRUCTURE
DUPLEXING REBUILD
METHOD : SYSTEM-MANAGED
REBUILD PHASE: DUPLEX ESTABLISHED
REALLOCATE EVALUATION PENDING
CQS_FF_LOGSTR 03/03/2004 08:26:06 DUPLEXING REBUILD OLD STRUCTURE
DUPLEXING REBUILD
CQS_FF_LOGTEST 03/03/2004 08:24:08 ALLOCATED
REALLOCATE EVALUATION PENDING

```

```

CQS_FP_LOGSTR 03/03/2004 08:26:10 DUPLEXING REBUILD NEW STRUCTURE
DUPLEXING REBUILD
METHOD : SYSTEM-MANAGED
REBUILD PHASE: DUPLEX ESTABLISH
REALLOCATE EVALUATION PENDING
CQS_FP_LOGSTR 03/03/2004 08:26:03 DUPLEXING REBUILD OLD STRUCTURE
DUPLEXING REBUILD
CQS_FP_LOGTEST 03/03/2004 08:24:10 ALLOCATED
REALLOCATE EVALUATION PENDING
CSLRMGR_PROD 03/03/2004 08:26:00 ALLOCATED
REALLOCATE EVALUATION PENDING
CSLRMGR_TEST 02/02/2004 15:59:24 ALLOCATED
REALLOCATE EVALUATION PENDING
DFHCFLS_CFDTA 03/02/2004 20:44:21 DUPLEXING REBUILD NEW STRUCTURE
DUPLEXING REBUILD
METHOD : SYSTEM-MANAGED
REBUILD PHASE: DUPLEX ESTABLISHED
REALLOCATE EVALUATION PENDING
:

```

(Additional log output omitted...)

```

WAS5P1_ERRLOG -- -- NOT ALLOCATED
WAS5T1_ERRLOG -- -- NOT ALLOCATED
WAS50TST_ERRLOG -- -- NOT ALLOCATED

```

Example: The following is an example of issuing the DISPLAY XCF,STR,STRNAME=COUPLE_CKPT1 command during REALLOCATE processing.

```

STRNAME: COUPLE_CKPT1
STATUS: REASON SPECIFIED WITH REBUILD START:
POLICY-INITIATED
DUPLEXING REBUILD
METHOD : SYSTEM-MANAGED
AUTO VERSION: BADC074A A28C45A1
REBUILD PHASE: DUPLEX ESTABLISHED
REALLOCATE EVALUATION PENDING
TYPE: SERIALIZED LIST
POLICY INFORMATION:
POLICY SIZE : 50560 K
POLICY INITSIZE: N/A
POLICY MINSIZE : 0 K
FULLTHRESHOLD : 95
ALLOWAUTOALT : NO
REBUILD PERCENT: N/A
DUPLEX : ENABLED

```

Example: The following is an example of issuing the SETXCF STOP,REALLOCATE command.

```

14:00:39.68 TP0D1 00000200 SETXCF STOP,REALLOCATE
14:00:39.79 00000000 IXC543I THE REQUESTED STOP,REALLOCATE WAS ACCEPTED

```

Example: The following is an example of issuing the SETXCF STOP,REALLOCATE,FORCE command.

```

17:43:19.08 IBMUSR8 00000200 SETXCF STOP,REALLOCATE,FORCE
SETXCF STOP,REALLOCATE,FORCE
IXC543I THE REQUESTED STOP,REALLOCATE,FORCE WAS ACCEPTED. 684
IXC545I REALLOCATE PROCESSING RESULTED IN THE FOLLOWING: 685
0 STRUCTURE(S) REALLOCATED - SIMPLEX
1 STRUCTURE(S) REALLOCATED - DUPLEXED
0 STRUCTURE(S) POLICY CHANGE MADE - SIMPLEX
0 STRUCTURE(S) POLICY CHANGE MADE - DUPLEXED
0 STRUCTURE(S) ALREADY ALLOCATED IN PREFERRED CF - SIMPLEX
2 STRUCTURE(S) ALREADY ALLOCATED IN PREFERRED CF - DUPLEXED
0 STRUCTURE(S) NOT PROCESSED
0 STRUCTURE(S) NOT ALLOCATED

```

```

3  STRUCTURE(S) NOT DEFINED
-----
6  TOTAL
0  ERROR(S) ENCOUNTERED DURING PROCESSING
IXC543I THE REQUESTED STOP,REALLOCATE,FORCE WAS COMPLETED. 686
    
```

Migrating to z/OS.e V1R5

This section describes our migration experiences with z/OS.e V1R5.

z/OS.e V1R5 base migration experiences

This section describes our experiences with migrating one system image (JH0) from z/OS.e V1R4 to z/OS.e V1R5. Here we only cover our experiences with our base migration to z/OS.e V1R5, including our high-level migration process and other migration activities and considerations.

Our high-level migration process for z/OS.e V1R5

The following is an overview of our z/OS.e V1R5 migration process.

Before we began: We reviewed the information in *z/OS and z/OS.e Planning for Installation*, GA22-7504, which covers both z/OS V1R5 and z/OS.e V1R5.

Important notice about cloning and software licensing

As discussed in *z/OS and z/OS.e Planning for Installation*, you might find that sharing system libraries or cloning an already-installed z/OS or z/OS.e system is faster and easier than installing z/OS or z/OS.e with an IBM installation package such as ServerPac. Most Parallel Sysplex customers are already aware of the concept of cloning and the benefits it provides.

However, prior to sharing or cloning z/OS or z/OS.e, **you must have a license for each z/OS and z/OS.e operating system that you run.** If you don't have the appropriate license or licenses, you must contact IBM. Any sharing or cloning of z/OS or z/OS.e without the appropriate licenses is not an authorized use of such programs. On a z800 server, if you want to run both z/OS and z/OS.e, z/OS requires the appropriate license for the machine on which it runs and z/OS.e requires a license for the number of engines on which it runs.

For more information about z/OS.e licensing, see *z800 Software Pricing Configuration Technical Paper* at www.ibm.com/servers/eserver/zseries/library/techpapers/pdf/gm130121.pdf.

Table 12 shows the high-level process we followed to migrate our z/OS.e V1R4 system to z/OS.e V1R5.

Table 12. Our high-level migration process for z/OS.e V1R5

Stage	Description
Obtaining licenses for z/OS.e	You need a license for the appropriate number of engines on the z800 server on which you intend to run z/OS.e (and, you would also need a license to run z/OS on the z800, if you intend to install it there). We use an internal process to do this; however, you must use the official process stated in <i>z800 Software Pricing Configuration Technical Paper</i> .

Table 12. Our high-level migration process for z/OS.e V1R5 (continued)

Stage	Description
Updating the z800 LPAR name	z/OS.e must run in LPAR mode and the LPAR name must be of the form ZOSExxxx, where xxxx is up to 4 user-specified alphanumeric characters. The name of the LPAR in which we run z/OS.e is ZOSEJH0. (We used HCD to set this when we first installed z/OS.e V1R3.)
Updating parmlib for z/OS.e V1R5	z/OS.e requires the LICENSE=Z/0SE statement in the IEASYSxx parmlib member. We used the same SYS1.PETR15.PARMLIB data set that we created for z/OS V1R5. We then have separate IEASYSxx and IFAPRDxx members in SYS1.PARMLIB that we tailored specifically for z/OS.e. See “Updating system data sets for z/OS.e” for details.
Updating our LOADxx member	During our initial installation of z/OS.e V1R3, we updated the entry for our system JH0 in our LOADxx member in SYS0.IPLPARM to point to our new IEASYS02 parmlib member and to reflect the new LPAR name. Therefore, we did not need to change it for V1R5.
Updating our IEASYMPT member	During our initial installation of z/OS.e V1R3, we updated the entry for our system JH0 in our IEASYMPT member in SYS1.PETR13.PARMLIB to point to our new IFAPRDxx parmlib member and to reflect the new LPAR name. Therefore, when we created our new SYS1.PETR15.PARMLIB, we carried the change along for V1R5.
IPLing the z/OS.e V1R5 image	We brought up z/OS.e V1R5 on our JH0 production system.

More about our migration activities for z/OS.e V1R5

This section highlights additional details about some of our migration activities.

About our z800 LPAR environment: z/OS.e must run in LPAR mode on a zSeries 800 mainframe server; it cannot run in basic mode. In addition, the name of the LPAR in which z/OS.e runs must be of the form ZOSExxxx, where xxxx is up to four user-specified alphanumeric characters. The name of our z/OS.e LPAR is ZOSEJH0.

Note: You can only run z/OS.e in a partition named ZOSExxxx. You cannot IPL a z/OS system in a partition named ZOSExxxx.

We currently run z/OS.e (JH0) in a mixed LPAR environment alongside LPARs running z/OS (JG0) and z/VM (PETVM2) on the same z800 server.

Note: Don't let the fact that z/OS.e only runs on a z800 server confuse you. A z800 is a fully functional zSeries server and, in addition to z/OS.e, it supports all of the same zSeries operating systems as a z900 or z990 server.

Updating system data sets for z/OS.e: We continue to use concatenated parmlib support to add or update parmlib members for z/OS.e V1R5. We use the same SYS1.PETR15.PARMLIB data set as we do for our z/OS V1R5 systems.

Below are examples of our parmlib customizations to accommodate z/OS.e V1R5. Appendix A, “Some of our parmlib members,” on page 305 summarizes the changes we made by parmlib member.

Example: We have a separate IEASYSxx member, IEASYS02, which specifies the LICENSE=Z/0SE statement that z/OS.e requires.

The entry for our z/OS.e system (JH0) in our LOADxx member in SYS0.IPLPARM points to our IEASYS02 parmlib member and specifies the name of our z/OS.e LPAR, as follows:

```

:
:
HWNAME      z800name
LPARNAME    ZOSEJH0
PARMLIB     SYS1.PETR15.PARMLIB
SYSPARM     02
:
:

```

Example: We have a separate IFAPRDxx member, IFAPRD02, which specifies the product ID value 5655-G52 for z/OS.e. There is no change to the product name value for z/OS.e (the product name value remains Z/OS).

Below is an example of one of the entries from our IFAPRD02 member:

```

:
:
PRODUCT OWNER('IBM CORP')
          NAME(Z/OS)
          ID(5655-G52)
          VERSION(*) RELEASE(*) MOD(*)
          FEATURENAME(Z/OS)
          STATE(ENABLED)
:
:

```

We also have an entry for our system JH0 in our IEASYMPT member in SYS1.PETR15.PARMLIB to point to our new IFAPRD02 parmlib member and to reflect the z/OS.e LPAR name, as follows:

```

:
:
SYSDEF HWNAME(z800name)
        LPARNAME(ZOSEJH0)
        SYSNAME(JH0)
        SYSCLONE(JH)
:
:
SYMDEF(&PROD='02')
:
:

```

Using current z/OS.e levels of JES2 and LE: As required, we are using the level of JES2 and Language Environment (LE) that comes with z/OS.e V1R5. z/OS.e does not permit the use of a lower level JES2 (or JES3) or LE.

Updating the ARM policy: You must ensure that your automation policies, such as ARM, do not try to use a z/OS.e image to start products that z/OS.e does not support. For example, do not identify a z/OS.e image as a restart target in a Parallel Sysplex that contains a mix of z/OS.e and z/OS images where the z/OS images run IMS, CICS, or DB2 with a requirement for CICS. CICS, IMS, or DB2 that uses CICS cannot restart on a z/OS.e image, but must restart on a z/OS image. If, for example, a CICS region attempts to start on z/OS.e, the region will start but the applications will fail with a U4093 abend.

Back when we installed z/OS.e V1R3, we removed our z/OS.e image, JH0, as a restart target for the unsupported subsystems mentioned above.

Removing z/OS.e from participation in MNPS: In our environment, CICS is the only exploiter of multiple node persistent sessions (MNPS) support. Because CICS cannot run on z/OS.e, there is no reason for the VTAM on z/OS.e to connect to the MNPS structure, ISTMNPS. We removed our z/OS.e image from participating in MNPS by coding the STRMNPS=NONE statement in our VTAM start member, ATCSTRxx, in SYS1.VTAMLST.

Removing z/OS.e from participation in TSO generic resource groups: Since TSO on z/OS.e only allows a maximum of eight concurrent sessions, we removed our z/OS.e image from participating in TSO generic resource groups. You can do this by coding the GNAME=NONE parameter—either in a separate TSOKEYxx member in parmlib or on the START command that starts the terminal control address space (TCAS).

In our case, we use a single TSOKEYxx member that has a symbolic value for the GNAME parameter. We then set that symbol to NONE for our JH0 image in our IEASYMPT member.

Other experiences with z/OS.e V1R5

Our testing of z/OS.e V1R5 included the following workloads or scenarios:

- z/OS UNIX System Services
- DB2 UDB
- IBM HTTP Server in scalable server mode
- WebSphere Application Server for z/OS
- CICS Transaction Gateway (CTG) to access CICS regions running in z/OS images on the same CPC and other CPCs
- DB2 access from Linux guests under z/VM on the same CPC
- our Bookstore application transactions

Using the IBM Health Checker for z/OS and Sysplex

The IBM Health Checker for z/OS and Sysplex is a tool that checks the current, active z/OS and sysplex settings and definitions for an image and compares their values to those either suggested by IBM or defined by the installation as the criteria. The objective of the Health Checker is to identify potential problems before they impact system availability or, in the worst cases, cause outages.

We are using Version 3 of the IBM Health Checker for z/OS and Sysplex, which we downloaded from the z/OS downloads page at www.ibm.com/servers/eserver/zseries/zos/downloads/. The documentation, *z/OS and Sysplex Health Checker User's Guide*, SA22-7931, is also available on this Web page.

Using the default USERPARM member supplied with the Health Checker, we created and customized several new members to perform various types of checking. For instance, we use one member with one set of parameters to perform XCF checks, another member with different parameters to perform APF and LINKLST checks, and so on. This also makes the reports easier to look at and use. It also allows us to isolate the checks that examine values that have a sysplex scope so that we can run them on only one z/OS image, rather than on every image in the sysplex—thereby eliminating redundant information from the reports for each image.

z/OS performance

The performance of our z/OS systems is an important issue for us, just as it is for you. If we are to be customer-like, we must pay attention to meeting the goals in our service level agreements.

The following describes what we do in each phase of our testing, and what we plan to periodically report to you in our test reports:

- Monitor our performance in terms of our service level agreements

Our goal for our sysplex workloads continues to be 90% CP utilization across the systems in the sysplex, with WLM goals such as 80% of CICS transactions completed in less than 0.6 seconds on those images where CICS runs. We fill in the remaining 10% with batch work and various additional types of users, such as z/OS UNIX users (such as WebSphere for z/OS), TSO users, and workstation clients.

Note: This is not formal performance testing for purposes of publishing performance statistics for z/OS. It is a way for us to establish and report on reasonable goals for response times and transaction rates for the various types of workloads we run, just as a customer would do to create a service level agreement (SLA).

- Identify performance problems in our environment, find solutions to those problems, and report the information to you.
- Provide you with periodic performance snapshots of our environment, in the form of RMF reports, to provide pertinent information such as how many transactions we process per second and what our response times are for various workloads. You can find those reports in Appendix B, “Some of our RMF reports,” on page 307.

Chapter 3. Using zSeries Application Assist Processors (zAAPs)

As promised, we've updated our most recent testing of zAAP since the June 2004 report.

IBM @server zSeries 890 (z890) and zSeries 990 (z990) servers make available a new, optional feature—the zSeries Application Assist Processor (zAAP)—which provides a strategic z/OS Java execution environment for customers who desire the powerful integration advantages and traditional qualities of service of the zSeries platform.

A zAAP is similar in concept to a System Assist Processor (SAP). Unlike CPs, ICFs, and IFLs, zAAPs can do nothing on their own; they cannot perform an IPL and cannot run an operating system. zAAPs 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 Java programming under control of the IBM Java Virtual Machine (JVM).

This chapter describes what we did to configure and to prepare to exercise the zAAP feature on our z990 server.

Prerequisites for zAPP

The following are prerequisites for zAAP and execution of the JVM processing cycles:

- the IBM Software Developer's Kit (SDK) for z/OS
- z/OS 1.6 (or z/OS.e 1.6)
- Java 2 Technology Edition V1.4.1 with PTF for APAR PQ86689
- and the Processor Resource/Systems Manager™ ((PR/SM™) must be enabled).

Subsystems and applications using SDK 1.4 that exploit zAAPs

The following subsystems and applications using SDK 1.4 exploit zAAPs:

- WebSphere Application Server 5.1
- CICS/TS 2.3
- DB2 V7, DB2 V8 (we tested with both)
- IMS V7, IMS V8, and IMS V9 (we tested with IMS V8)
- Websphere MQ 5.3.1
- Websphere Business Integration Message Broker 5.0

Setting up zAAP

First we executed the recommended zAAP Projection Tool for Java 2 Technology Edition SDK 1.3.1 against the available Java workload we had at the early stage of our testing. This provided us a baseline or a starting point for defining the minimum number of zAAP processors we would require on our z990 and z890 processors. Please reference the following for more details pertaining to the Projection Tool:

- “Installation of the zAAP Projection Tool Instrumented SDK in WebSphere for z/OS Version 5” available at:

<http://www-1.ibm.com/support/techdocs/atmastr.nsf/WebIndex/WP100431>

- z/OS Performance: Capacity Planning Considerations for zAAP Processors available at:

<http://www-1.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP100417>

As mentioned in the intro we recommend that you contact your hardware support for the latest hardware and software requirements. We licensed 1 CP on our z890 and 2 CPs on our z990 as zAAPs.

Configuring zAAPs

We configured two zAAPs on all our z/OS images on our z990 server and we configured one zAAP on our z890 server. When you configure the z/OS logical partitions, you simply specify how many logical zAAPs you want to configure for each partition, just as you do the number of standard CPs. When you IPL the system, z/OS determines how many zAAPs are configured and manages an additional dispatcher queue for zAAP-eligible work.

We did the following to configure the zAAPs:

1. Updated the image profiles for the Z2 and Z3 partitions to define two zAAPs to each partition.

Example: Figure 5 shows an example of the image profile for our Z2 image with two zAAPs defined.

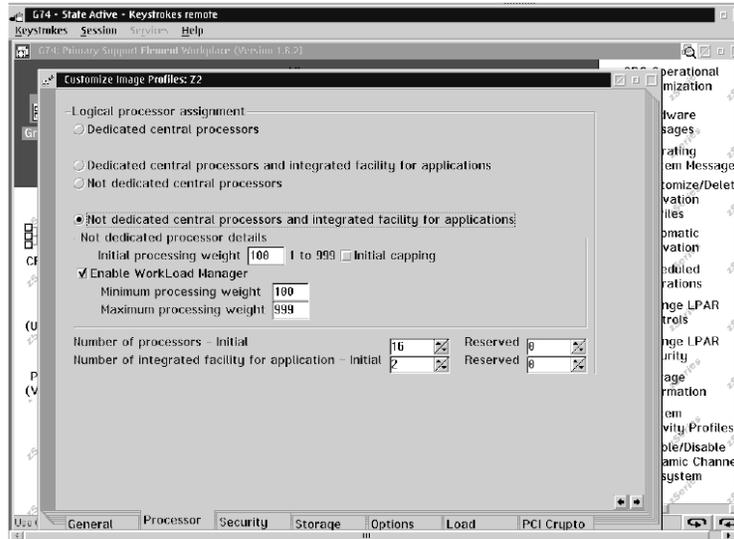


Figure 5. Example of the image profile for our Z2 image with two zAAPs defined.

2. Updated parmlib member IEAOPTxx for the z/OS partitions to specify the following options:

IFACROSSOVER=YES

zAAP-eligible work may execute on zAAPs or it can “cross over” and execute on standard processors.

IFAHONORPRIORITY=YES

Standard processors execute both Java and non-Java work in order of dispatching priority.

3. Deactivated and reactivated the z/OS partitions to bring the zAAPs online.

You can use the D M=CPU command to display the status of the zAAPs. The zAAPs appear as assist processors in the response to the D M=CPU command.

Example: The following is an example of the response to the D M=CPU command on system Z2:

```
IEE174I 15.34.28 DISPLAY M
PROCESSOR STATUS
ID CPU SERIAL
00 + 02B52A2084
01 + 02B52A2084
02 + 02B52A2084
03 + 02B52A2084
04 +A 02B52A2084
05 +A 02B52A2084

CPC ND = 002084.D32.IBM.00.00000001B52A
CPC SI = 2084.325.IBM.00.000000000001B52A
CPC ID = 00
CPC NAME = G74
LP NAME = Z2 LP ID = 2
CSS ID = 0
MIF ID = 2
```

Example: The following is an example of the response to the D M=CPU command on system Z3:

```
IEE174I 15.38.30 DISPLAY M
PROCESSOR STATUS
ID CPU SERIAL
00 + 24B52A2084
01 + 24B52A2084
02 + 24B52A2084
03 + 24B52A2084
04 +A 24B52A2084
05 +A 24B52A2084

CPC ND = 002084.D32.IBM.00.00000001B52A
CPC SI = 2084.325.IBM.00.000000000001B52A
CPC ID = 00
CPC NAME = G74
LP NAME = Z3 LP ID = 24
CSS ID = 2
MIF ID = 4
```

Monitoring zAAP utilization

There is support in RMF (supplied by APAR OA05731) to provide information about zAAP utilization. This information is useful to determine if and when you need to add additional zAAP capacity.

SMF is another source of information. SMF type 72 records contain information about zAAP utilization. There are also new fields in SMF type 30 records to indicate

the amount of time spent on zAAP work as well as the amount of time spent executing zAAP-eligible work on standard processors for both crossover and honor priority execution modes.

Here is an example of our RMF Monitor III displaying the use of the zAAPs on our z990 processor highlighted in **bold**:

```

RMF V1R5  CPC Capacity                               Line 1 o
Command ==>                                         Scroll ==>
Switched to option set WLMPOLO1 on JA0.
Samples: 120   System: JA0   Date: 09/14/04   Time: 11.09.00   Range: 120

Partition:  JA0           2084 Model 325
CPC Capacity:  1114   Weight % of Max: 10.0       4h MSU Average:  175
Image Capacity: 1069   WLM Capping %:  ****       4h MSU Maximum:  205

Partition  --- MSU --- Cap Proc   Logical Util %   - Physical Util % -
           Def  Act  Def  Num    Effect  Total   LPAR  Effect  Total

*CP
EBTELNX      0    1  NO   2.0     1.3    1.3    0.0    0.1    0.1
JA0           0   191 NO   7.0    60.3   61.2    0.3   16.9   17.1
JC0           0   158 NO   7.0    49.8   50.6    0.2   13.9   14.2
JE0           0   137 NO   8.0    37.9   38.6    0.2   12.1   12.3
Z2           0    27 NO   7.0     8.5    8.7    0.1    2.4    2.4
Z3           0    17 NO   4.0     9.3    9.5    0.0    1.5    1.5
PHYSICAL                                1.3                                1.3

*ICF
CF2                                3.0    98.9   98.9    0.0   42.4   42.4
JA0                                NO   2.0   46.7  46.9  0.1  13.3  13.4
JC0                                NO   2.0   0.0   0.0  0.0  0.0  0.0
JE0                                NO   2.0   0.0   0.0  0.0  0.0  0.0
PETVM                                NO    2.0   15.0   15.6    0.2    4.3    4.4
Z2                                NO   2.0   2.0   2.1  0.0  0.6  0.6
Z3                                NO    2.0    0.1    0.1    0.0    0.0    0.0
PHYSICAL                                1.0                                1.0

```

Preparing our workloads to exercise the zAAP feature

Initially, we selected several of our current MQ Web workloads and rewrote them as base Java applications (non-Web) to exercise the zAAP feature. We also installed WebSphere Business Integration Message Broker 5.0 for zAAP testing. This product itself uses Java and will increase the utilization of the zAAP feature in addition to the workloads. .

In the near future, we plan to use the following MQ-based workloads (which run via TPNS scripts and TSO users) to test the zAAP feature:

- MQLARGE — This workload currently runs primarily on system JG0. Its purpose is to create temporary MQ queues and put large messages on them. We added a compute module to increase the application's CPU usage, as MQ itself does not consume much CPU resource.
- MQCICS — This workload runs on system JB0 and puts a request message on a CICS bridge queue to run a DB2 transaction.
- MQDQM — This is a communications test workload that puts and gets messages between a local MQ queue manager and a remote system's queue manager.
- MQDQLSSL — This workload is similar to the MQDQM workload but uses SSL channels to test security.

- RetailTPNS — This workload simulates a retail type of application and uses WebSphere MQ Integrator to put messages on a queue for the broker to process. We run a TPNS version in Java that will use the zAAP feature.

Other workloads we support are:

- DB2— In the past, Integration Test had implemented the NST (Native Stress Test) Version 6 workload for DB2, which is a TPNS driven, CICS based workload comprised of COBOL programs, stored procedures and user defined functions (UDFs). For z/OS 1.6 with the introduction of eServer™ zSeries Application Assist Processor, several of the stored procedures originally written in COBOL were converted to Java, and a new Workload Manager (WLM) address space to run the Java stored procedures was defined.
- IMS— The IMS Java workload is based on the IMS Java Dealership IVP application, which is documented in the "IMS Java User's Guide" (SC27-1296-00). The environment is IMS V8 running SDK 1.4.2. This application consists of one database and one transaction (with multiple methods). We modified the IMS V8 Dealership IVP application by:
 - Creating MFS screen formats so the transactions can be set up as an OLTP workload
 - Coded TPNS scripts to drive different dealership application methods.

We are currently running the following methods: FindACar, ListModels and ShowModelDetails.

We executed our zAAP testing with different configurations. The majority of our testing was completed with our zAAPs configured online. We also performed extensive testing with the zAAP processors configuring them online and offline with Java workload running.

Chapter 4. Migrating to CICS TS Version 2 Release 3

In this section, we describe our experiences with migrating to CICS Transaction Server Version 2 Release 3 (CICS TS 2.3). CICS TS 2.3 contains the following:

- HBDD110 CICS Application Migration Aid
- HCI6300 CICS - Base
- HCP2300 CICS/Plex System Manager - Base
- H0B5110 REXX/CICS Runtime
- H0B7110 REXX/ CICS Development
- H0Z2110 REXX/CICS Common
- JCI630D CICS - JAVA/IIOP
- JCI6301 CICS - COBOL feature
- JCI6302 CICS - PL/1 feature
- JCI6303 CICS - 'C' feature
- JCP2302 CICS/Plex System Manager - SASC feature

Applicable documentation: During the migration to CICS TS 2.3, we used the following publications:

- *CICS Transaction Server for z/OS Program Directory*, GI10-2560
- *CICS Transaction Server for z/OS Migration from CICS TS Version 2.2*, GC34-6223
- *CICS Transaction Server for z/OS Installation Guide*, GC34-6224
- *CICS Messages and Codes*, GC34-6241
- *CICS Operations and Utilities Guide*, SC34-6229
- *CICSplexSM for CICS Transaction Server for z/OS V2.3 Messages and Codes*, GC34-6265

Overview of migrating to CICS TS 2.3

As in the past, our goal with this migration was to follow the path of a typical customer. However, this time we migrated more slowly across the sysplex, and within the workloads on the sysplex. This created a thorough mixture of releases on a system as well as across the CICSplex. We did this to uncover as many compatibility and operational problems as possible. After we tested the migration steps on our test CICSplex, we were ready to migrate our production CICSplex.

In the last year, we've added another workload to our CICSplex. In addition to the original data-sharing workloads (IMS, VSAM-RLS, DB2), we now have a fourth workload, used primarily to test new z/OS Cryptographic hardware, ICSF, and Java. We call this our application group-C workload, which exploits the CICS-ICSF attach facility and the CICS-Java interface. In the future, as we develop more applications, we plan to add them to this group so we will have a more diversified application group.

Figure 6 on page 60 shows our CICS TS 2.3 and CPSM 2.3 latest configuration:

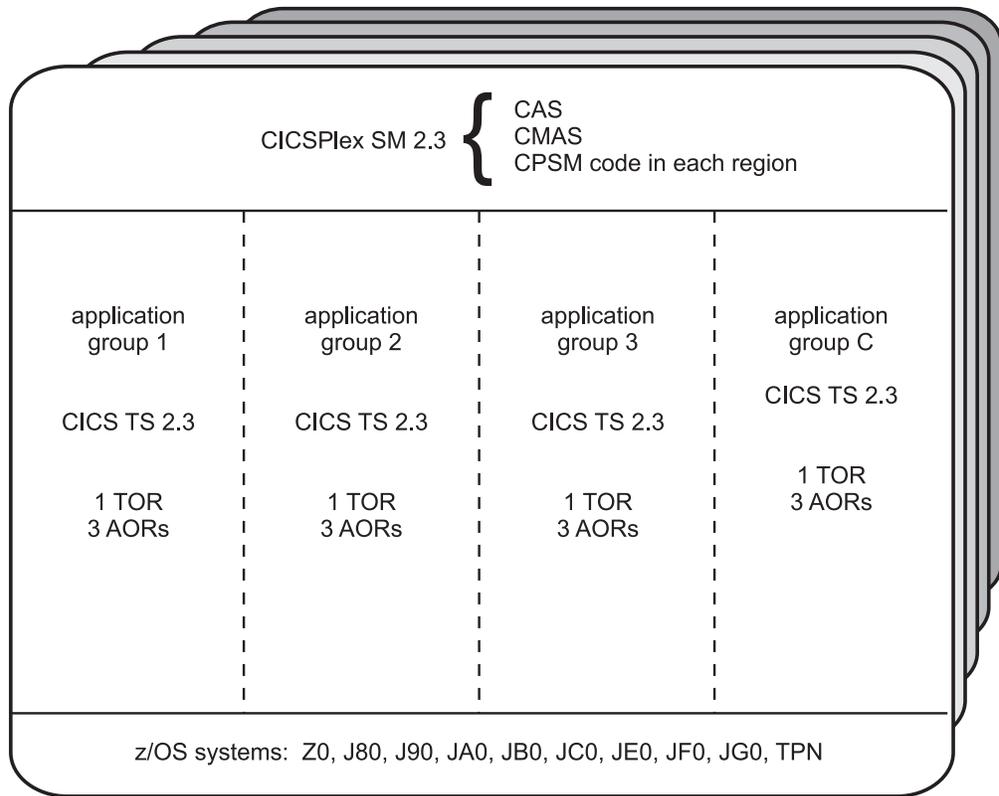


Figure 6. Our CICS TS 2.3 and C/PSM 2.3 configuration

When all of our systems are up and running we have a total of 10 CASs/CMASs monitoring 118 CICS regions (MASs). In one month we had approximately 50% of the CICSplex migrated. We had CTS22 CMASs communicating with CTS23 CMASs. Under the CTS23 CMASs, we had a mixture of CTS22 and CTS23 MASs. We did find a couple of migration related problems, as well as some general CMAS communications type problems. See “Experiences with migrating to CICS TS 2.3” on page 63.

At this point we stopped the migration, for about a month, to further expose the CICSplex to the z/OS testing already in progress, and to test fixes for problems we found. In the following month, we completed the migration.

Performing the migration to CICS TS 2.3

This section describes how we migrated to CICS TS 2.3.

Preparing for migration

Before we began the actual migration process, we did some preparatory work in the following areas:

Backing up our data: Even though we created new files and data sets, as a precaution, we first took backups of all of the CSDs and data repositories.

Alias’s: We defined new aliases for CTS23.

Loading the CTS23 product libraries: We set up and ran the SMPE jobs to load the CTS23 product libraries. We then ran a copy job to bring the build libraries over to our production systems.

Allocating supporting PDS's: We created copies of all our supporting libraries (JCL, SYSIN, tables, and so on). We reviewed these and updated accordingly with all the necessary CTS23 changes.

Customizing the CICS region data sets: We customized the jobs in *hlq.SDFHINST(DFHDEFDS)* and *hlq.SEYUINST(EYUDEFDS)* to define all of the region data sets and submitted them a number of times. Depending on your environment, you might need to alter the default file sizes. We increased the file sizes for the DFHGCD, DFHINTRA and DFHTEMP data sets.

Reviewing and reassembling tables: We reviewed and reassembled any tables we had modified.

Updating SYS1.PARMLIB and APF-authorizing program libraries: In SYS1.PARMLIB, we updated LINK list and LPA list. We APF-authorized the following program libraries:

- *hlq.SDFHAUTH*
- *hlq.SDFHLINK*
- *hlq.SEYUAUTH*
- *hlq.SEYULINK*

In PROCLIB, we reviewed and updated all our procs for the CTS23 changes.

Note: Remember, DFHIRP must be at the highest level of the code.

Migrating CICSplex SM

If you are migrating to CICSplex SM for the first time, CPSM consists of the following parts on each system:

- CAS (coordinating address space)
- CMAS (CICS-managed address space)
- CPSM code running in each CICS region (MAS), which communicates with the CMAS, sometimes referred to as the "agent" code. On any specific z/OS system, all three parts of CPSM must be at the same release level. As long as all CPSM components on any single z/OS system are at the same level, you can run mixed levels of CPSM on different systems within a sysplex.

Migrating the CASs

We reviewed the steps documented in *CICS Transaction Server for z/OS Migration Guide* to migrate the CASs.

Steps for migrating the CASs

We did the following to migrate the CASs:

1. Defined a new BBIPARM parameter repository data set for CPSM 2.3 We reviewed the JCL in the EYUDEFDS member, customized the JCL statements that allocate the CAS parameter library (EYUIPARM) data set, and then submitted it. The BBIPARM DD name contains the cross-system definitions for CPSM.

Note: CASs running at different levels cannot share the same BBIPARM data set.

2. Updated our TSO signon procedures to point to the new data sets for the new release of CPSM 2.3

3. Reviewed the JCL in the EYUCAS member for any changes to the CAS startup procedure.
Because there were no changes, we simply made a copy of our old procedure and updated the data set names to use our new high-level qualifiers. (The EYUCAS member resides in the *hlq*.SEYUINST library.)

4. Started the CAS

5. Defined the CAS and updated the parameter repository (BBIPARM), as follows:
 - a. From the TSO EUI address space (CPSM), selected option 1 to invoke the PLEXMGR view
 - b. Invoked the CASDEF view, which put us into the browse mode
 - c. Entered the EDIT command to change to edit mode
 - d. Entered the C action command to select our CAS, which took us to the **Change CAS System Definition** panel
 - e. Made the appropriate changes to our environment

Note: When the CAS first comes up, it takes a default group name of EYUGR230. We changed our XCF group name to EYUGP230 for our production CAS's. (EYUGT230 for test). Remember, as in any migration, CAS's running at different release levels cannot communicate with each other.

Migrating the CMASs

Steps for migrating the CMASs

We did the following to migrate the CMASs:

1. Defined a new CSD

2. Updated the CSD with CPSM 2.3 level resource definitions and the CICS startup group list
We did this by running the DFHCSDUP utility with the UPGRADE command, as discussed in *CICS Operations and Utilities Guide*.

3. Updated the CICS system initialization table (SIT) overrides as follows:
 - changed GRPLIST parameter to point to the new CPSM 2.3 group list. EYU230L0
 - added WRKAREA=2048, per the migration guide

4. Reviewed our CICS resource definition tables, which we updated earlier

-
5. Converted the CPSM data repository to the CPSM 2.3 level by running the EYU9XDUT utility, as discussed in *CICS Transaction Server for z/OS Installation Guide*

 6. Reviewed the JCL in the EYUCMAS member for any changes to the CMAS startup procedure.
Because there were no changes, we simply made a copy of our old procedure and updated the data set names to use our new high-level qualifiers.

 7. Updated the MAS JCL to point to the new CPSM data sets, in order to identify the new CMAS code to the MAS regions.

Our CMASs were then ready to start.

Migrating the MASs

We reviewed the steps documented in *CICS Transaction Server for z/OS Migration Guide* to migrate the MASs. Many of the steps are similar to the steps we followed to migrate the CASs and CMASs.

Steps for migrating the MASs

We did the following to migrate the MASs:

1. Defined a new CSD and copied our application groups from the old CSD

2. Upgraded the CSD using "UPGRADE USING(EYU963G1)" for CPSM 2.3. We also removed groups for previous releases of CPSM from the group list

3. Reviewed our CICS resource definition tables, which were updated earlier

4. Copied the JCL for our MAS startup procedure and changed the library names to use our new high-level qualifiers;
Reviewed the LE libraries we had in RPL concatenation.

5. Made Java changes. In CICS TS 2.2, JVM profiles were stored in a PDS member. In CICS TS 2.3 they are stored in the HFS directory pointed to by the JVMPROFILEDIR system initialization parameter. We removed the DFHJVM DD card in the CICS start up proc.

Note: We keep our JVM profiles outside the HFS shipped with CTS23, so that they would not be overridden with a CTS23 HFS at maintenance time.

Our MASs were then ready to start.

Experiences with migrating to CICS TS 2.3

This section describes the types of problems we encountered and their solutions.

| **Install problems:** After we had migrated a couple of our CMASs and MASs to
| CTS23, we found that certain CPSM functions and views did not work correctly. For
| example, when we tried a hyperlink from the CICSSYS group, our CPSM view
| hung. Another example was when we cleared TRANDUMP. When we were logged
| onto a CMAS at the higher-code level, and we able to clear TRANDUMP. But if we
| logged onto a CMAS at the lower level, we received the following error:

| EYUEI0574E Action cannot be performed for scope.

| Both these problems are fixed by: APAR *PQ89462* - PTF *UQ89744*.

| Another problem with the mixed releases, was that we could not see all the regions
| in the CICSRGNS view, for any MASs at the higher levels. This problem is fixed by
| APAR *PQ89791* - PTF *UQ89745*, and *UQ89746*.

| Again, as result of mixed release, after we had restarted some TORs, they failed to
| joined CPSM. When we checked we found the following errors in the MAS logs:
| +EYUXL0138I EYU9XLOP entering wait for workload.

| The CMAS log had the following:

| EYUWI0082E CMASJB0 WLM Workload query process failed for context (PETPLEX)
| - directed to CMAS (CMASJC0)
| EYUWM0401E CMASJB0 Workload Specification (G3WLM) failed to install for
| context(PETPLEX) - initiated by join
| EYUWM0401E CMASJB0 of router (CICS3TBA)

| This problem is fixed by APAR *PQ89798* - PTF *UQ91147*

| On one occasion, during the migration we ran into the following problem. After the
| CMASs had been restarted, the system hung while navigating through some CPSM
| views. The problem was diagnosed to be a repository-sync that never completed
| and is fixed by APAR *PQ90197* - PTF *UQ92562* / *UQ92563*

| **Note:** This problem was later determined to be a base CPSM problem, NOT
| related to migration.
|

Chapter 5. Migrating to DB2 Version 8

This chapter addresses the processes and experiences encountered during the migration of the Integration Test production 12 way DB2 data sharing group DB1G from DB2 Version 7 to Version 8 (composed of members DBA1, DBB1, DBC1, DBD1, DBE1, DBF1, DBG1, DBH1, DBI1, DBZ1, DB81, and DB91).

We used the *DB2 Installation Guide* 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*.

Migration considerations

Before you migrate to DB2 Version 8, note the following points:

- Migrations to DB2 Version 8 are only supported from subsystems currently running DB2 Version 7; 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 8 subsystem after it is in compatibility mode. For this step, only the following selected Version 7 IVP jobs can be executed:
 1. Version 7 phase 2 IVP applications
 - a. DSNTMJ2A - All steps except the first two
 - b. DSNTMJ2C - Only step PH02CS04, statement RUN PROGRAM(DSN8BC3) PLAN(DSN8BH61), is to be executed
 - c. DSNTMJ2D - Only step PH02DS03, statement RUN PROGRAM(DSN8BD3) PLAN(DSN8BD61), is to be executed
 - d. DSNTMJ2E - Only step PH02ES04, statement RUN PROGRAM(DSN8BE3) PLAN(DSN8BE61), is to be executed
 - e. DSNTMJ2F - Only step PH02FS03, statement RUN PROGRAM(DSN8BF3) PLAN(DSN8BF61), is to be executed
 - f. DSNTMJ2P - Execute step PH02PS05
 2. Version 7 phase 3 IVP applications
 - a. ISPF-CAF applications, with the exception of DSNTMJ3C and DSNTMJ3P.

If you want to run these IVPs as part of the verification of DB2 Version 8 compatibility mode, they must first be run under Version 7 in their entirety before you start the Version 8 migration process and must remain available for use after you complete the migration to Version 8 compatibility mode.

- Before you begin the migration process to DB2 Version 8, verify that if you are using CFCC levels 7 or 8, they are at the proper service levels - 1.06 and 1.03, respectively; to avoid the possibility of data corruption. Our three CFs are all running higher levels of service, as shown below:

```
CF1: CFCC RELEASE 14.00, SERVICE LEVEL 00.14
CF2: CFCC RELEASE 14.00, SERVICE LEVEL 00.14
CF3: CFCC RELEASE 13.00, SERVICE LEVEL 04.08
```

Other than these requirements, no other CFCC service level requirements exist.

- Examining "Migration Considerations" of the *DB2 Installation Guide*, the following items are of particular interest:
 - Global temporary tables require a 16K buffer pool.

- Declared temporary tables require at least one table space to have a page size of 8K or greater in the temporary database.
- Support for DB2-established data spaces for cached dynamic statements is removed; you can no longer specify parameters EDMDSPAC and EDMDSMAX during migration.
- Consider increasing IDBACK and CTHREAD subsystem parameters -- utilities might now require additional threads.
- Support for DB2-established stored procedure address spaces is removed (that is you can no longer specify NO WLM ENVIRONMENT when creating or altering a stored procedure); existing stored procedures can continue to run in a DB2-established stored procedure address space, but you should migrate such procedures to a WLM environment as soon as possible.
- A default DSNHDECP module (found in SDSNLOAD) is no longer shipped with DB2. DSNTIJUZ must be used to create a customized DSNHDECP, which is then placed in SDSNLOAD and SDSNEXIT.
- During the migration of the first member of a data sharing group to DB2 Version 8, 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 8 libraries exclusively.

Premigration activities

Before migrating to DB2 Version 8, application of the fallback SPE to all members of the Version 7 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 8 concatenations before invoking the DB2 installation CLIST:

- DB2.DB2810.SDSNSPFM was concatenated to ISPMLIB.
- DB2.DB2810.SDSNSPFP was concatenated to ISPPLIB.
- DB2.DB2810.SDSNSPFS was concatenated to ISPSLIB.
- DB2.DB2810.SDSNSPFT was **not** concatenated to ISPTLIB, as DB2 online help was not installed.

In some instances it might be necessary to issue the following RACF command for the SDSNLOAD library:

```
ralter program * addmem('h1q.SDSNLOAD'/'*****/NOPADCHK)
```

This might prevent error messages like the following:

```
CEE3518S The module DSNAOCLI was not found in an authorized library.
CSV042I REQUESTED MODULE DSNAOCLI NOT ACCESSED. THE MODULE IS NOT PROGRAM CONTROLLED.
```

For our setup, we issued the RACF command on behalf of LDAP.

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.db2810.sdsnc1st(dsntinst)' from ISPF option 6.
```

We filled in the first panel DSNTIPA1 as shown in Figure 7:

```

Session G - DB2 V8 Migration
File Edit View Communication Actions Window Help
DSNTIPA1 DB2 VERSION 8 INSTALL, UPDATE, MIGRATE, AND ENFM - MAIN PANEL
===> _

Check parameters and reenter to change:

 1 INSTALL TYPE          ==> migrate  Install, Update, or Migrate
                               or ENFM (Enable New Function Mode)
 2 DATA SHARING         ==> yes      Yes or No (blank for Update or ENFM)

Enter the data set and member name for migration only. This is the name used
from a previous Installation/Migration from field 7 below:
 3 DATA SET(MEMBER) NAME ==> db2.db2710.sdsnsamp(dsntidd1)

Enter name of your input data sets (SDSNLOAD, SDSNHACS, SDSNSAMP, SDSNCLST):
 4 PREFIX                ==> db2.db2810
 5 SUFFIX                 ==>

Enter to set or save panel values (by reading or writing the named members):
 6 INPUT MEMBER NAME     ==> DSNTIDXA  Default parameter values
 7 OUTPUT MEMBER NAME    ==> dsntidd1  Save new values entered on panels

PRESS: ENTER to continue  RETURN to exit  HELP for more information

MA  g 02/007

```

Figure 7. DSNTIPA1

When we pressed enter, the pop-up screen DSNTIPP2 appeared as shown in Figure 8 on page 68:

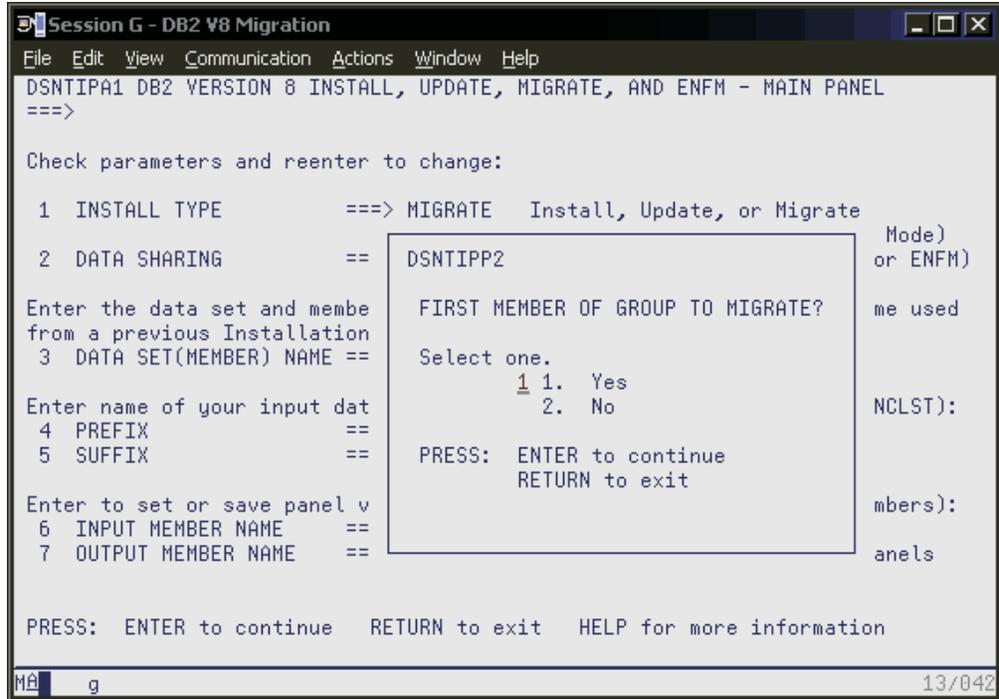


Figure 8. DSNTIPP2

We entered '1' to reflect that this was the first member of the data sharing group to be migrated to DB2 Version 8. From this point, we scrolled through the panels and accepted the existing values; upon completion, we placed the tailored CLISTs in DB2.DB2810.DBD1.SDSNSAMP and DB2.DB2810.NEW.SDSNTEMP, as shown in Figure 9 on page 69.

```

Session G - DB2 V8 Migration
File Edit View Communication Actions Window Help
DSNT478I BEGINNING EDITED DATA SET OUTPUT
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJHY)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJIN)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJTC)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJTM)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJIC)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJVC)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJSG)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJEX)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJGF)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJFT)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2810.NEW.SDSNTEMP(DSNU)', CLIST
DSNT489I CLIST EDITING 'DB2.DB2810.NEW.SDSNTEMP(DSNH)', CLIST
DSNT489I CLIST EDITING 'DB2.DB2810.NEW.SDSNTEMP(DSNHC)', CLIST
DSNT489I CLIST EDITING 'DB2.DB2810.NEW.SDSNTEMP(DSNEMC01)', CLIST
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJCX)', MIGRATE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJFY)', FALL BACK JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJUZ)', INSTALL JCL
***
g 19/006

```

Figure 9. Tailored CLISTs placed in DB2.DB2810.DBD1.SDSNSAMP and DB2.DB2810.NEW.SDSNTEMP

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 7 are still available for use. Failure to do so will prevent verifying that a successful migration to DB2 Version 8 compatibility mode has been made.
- Ensured that no utilities are running before migrating to DB2 Version 8. When the migration to Version 8 compatibility mode has been completed, any outstanding utilities that were started under Version 7 cannot be restarted or terminated under Version 8.
- EBCDIC and ASCII CCSID values must be nonzero. Note that this issue was addressed by several DB2 Version 7 PTFs (such as UQ74294), so ensure that the maintenance level for DB2 Version 7 is current before migrating to DB2 Version 8.

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. Before executing, we had to stop the following table spaces (through option 7, DB2 Commands of the DB2I Primary Option Menu):

```

DSNDB06.SYSDBASE
DSNDB06.SYSDBAUT
DSNDB06.SYSGROUP
DSNDB06.SYSPLAN
DSNDB06.SYSVIEWS
DSNDB01.DBD01

```

DSN1CHKR was then executed without incident. The table spaces were restarted which had been stopped.

DB2 recommends running DSN1COPY with the CHECK option on all of the catalog and directory table spaces. To accomplish this, we created a job called CKDIRCAT and ran the job with DB2 down. No problems occurred and we brought DB2 back up.

Next, we ran the CHECK index utility against all catalog and directory indexes (we created a job called CKIDRCAT). Again, no major problems occurred (we received a RC = 4).

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 8. 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 10 on page 71.

- **Making DB2 CLISTS available to TSO and batch users: DSNTIJVC** - Our logon proc QMFPROC must again be updated to add DB2.DB2810.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 QMFPROC), 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 update logon proc QMFPROC to add DB2.DB2810.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 QMFPROC to reflect the concatenation of the DB2 English DB2I panels as follows:
 - DB2.DB2810.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 instructed us to stop all DB2 V7 activity or else fallback procedures can fail. Before stopping data sharing group DB1G, we insured that there were no incomplete utilities (**@DBD1 DISPLAY UTILITY(*)**), and that no databases were in restrict or advisory status (**@DBD1 DISPLAY DATABASE(*) SPACE(*) RESTRICT** and **@DBD1 DISPLAY DATABASE(*) SPACE(*) ADVISORY**, respectively); brought down all members of DB1G.

We skipped optional **Migration Step 10 (Back Up your DB2 Version 7 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.

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

Migration Step 13 defines DB2 V8 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 performed previously.
- Step RENAME renames the current DB2 procedures in proclib. We skipped this step, however. The DB2 startup procs for DBD1 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.DB2810.PROCLIB.

We renamed the startup procs for DBD1 that reside in PET.PROCLIB (as per the RENAME step of DSNTIJMV). Next, we copied the new V8 startup procs for DBD1 from DB2.DB2810.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 V8 early code, we skipped **Migration Step 16**.

For **Migration Step 17**, the DBD1 member of data sharing group DB1G was started successfully. As shown in Figure 11, the level of the data sharing group is now 810 and in compatibility mode (**MODE(C)**). The DB2 level of DBD1 reflects that it is now running DB2 Version 8 code. The DISPLAY GROUP command shows the data sharing group is in compatibility mode and one member is running DB2 Version 8.

```

Session G - DB2 V8 Migration
-----
Display Filter View Print Options Help
SDSF OUTPUT DISPLAY DBD1MSTR S0047064 DSID 2 LINE 72 COLUMNS 17- 96
COMMAND INPUT ==> SCROLL ==> PAGE
*** BEGIN DISPLAY OF GROUP(DSNDB1G ) GROUP LEVEL(810) MODE(C)
                PROTOCOL LEVEL(1) GROUP ATTACH NAME(DB1G)
-----
DB2
MEMBER  ID  SUBSYS  CMDPREF  STATUS  DB2 SYSTEM  IRLM  IRLMPROC
-----
DBA1    4  DBA1    @DBA1    QUIESCED 710 JA0    IRA1  DBA1IRLM
DBB1    7  DBB1    @DBB1    QUIESCED 710 JB0    IRB1  DBB1IRLM
DBC1    6  DBC1    @DBC1    QUIESCED 710 JC0    IRC1  DBC1IRLM
DBD1    5  DBD1    @DBD1    ACTIVE   810 J90    IRD1  DBD1IRLM
DBE1    3  DBE1    @DBE1    QUIESCED 710 JE0    IRE1  DBE1IRLM
DBF1    2  DBF1    @DBF1    QUIESCED 710 JF0    IRF1  DBF1IRLM
DBG1   10  DBG1    @DBG1    QUIESCED 710 JC0    IRG1  DBG1IRLM
DBH1   11  DBH1    @DBH1    QUIESCED 710 JB0    IRH1  DBH1IRLM
DBI1   12  DBI1    @DBI1    QUIESCED 710 JE0    IRI1  DBI1IRLM
DBZ1    1  DBZ1    @DBZ1    QUIESCED 710 Z0    IRZ1  DBZ1IRLM
DB81    9  DB81    @DB81    QUIESCED 710 J80    IR81  DB81IRLM
DB91    8  DB91    @DB91    QUIESCED 710 J90    IR91  DB91IRLM
-----
SCA STRUCTURE SIZE: 9216 KB, STATUS= AC, SCA IN USE: 20 %
MA g 04/021

```

Figure 11. DISPLAY GROUP command

Up on deck next is CATMAINT, as outlined in **Migration Step 18**. We submitted and ran DSNTIJTC successfully. The job periodically issued message DSNU777I in SYSPRINT to indicate migration progress, as shown in Figure 12 on page 74:

```

Session G - DB2 V8 Migration
DSDF OUTPUT DISPLAY DSNTIJC J0047073 DSID 102 LINE 0 COLUMNS 02- 133 SCROLL ==> PAGE
COMMAND INPUT ==>
***** TOP OF DATA *****
DSNU000I DSNUGITC - OUTPUT START FOR UTILITY, UTILID = RELOCAT
DSNU1044I DSNUGTIS - PROCESSING SYSIN AS EBCDIC
DSNU050I DSNUGITC - CATMAINT UPDATE
DSNU750I DSNUEC00 - CATMAINT UPDATE PHASE 1 STARTED
DSNU777I DSNUEC00 - CATMAINT UPDATE STATUS - VERIFYING CATALOG IS AT CORRECT LEVEL FOR MIGRATION.
DSNU777I DSNUEC00 - CATMAINT UPDATE STATUS - BEGINNING MIGRATION SQL PROCESSING PHASE.
DSNU777I DSNUEC00 - CATMAINT CHECK STATUS - BEGINNING SYSDBASE TABLE SPACE MIGRATION PROCESSING.
DSNU777I DSNUEC00 - CATMAINT UPDATE STATUS - BEGINNING ADDITIONAL CATALOG UPDATES PROCESSING.
DSNU777I DSNUEC00 - CATMAINT UPDATE STATUS - PROCESSING SYSSTRINGS TABLE UPDATES.
DSNU777I DSNUEC00 - CATMAINT UPDATE STATUS - UPDATING DIRECTORY WITH NEW RELEASE MARKER.
DSNU752I DSNUEC00 - CATMAINT UPDATE PHASE 1 COMPLETED
DSNU777I DBD01 DSNUEC05 - CATMAINT UPDATE STATUS - CCSID UPDATES COMPLETED.
DSNU010I DSNUEC00 - UTILITY EXECUTION COMPLETE, HIGHEST RETURN CODE=0
***** BOTTOM OF DATA *****

```

Figure 12. Message DSNU777I displays CATMAINT progress

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

- 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.
- Ran DSN1CHKR as in **Migration Step 2** to ensure there were no broken links and that it ran successfully.
- Examined the queries in SDSNSAMP member DSNTESQ for discrepancies; none found.
- Ran DSN1COPY with the CHECK option on the catalog and directory table spaces. The job completed with a return code of 4, the result of step SYSDBASE, which produced the following report:

```

DSN1999I START OF DSN1COPY FOR JOB CKDIRCAT STEP1 SYSDBASE
DSN1989I DSN1COPY IS PROCESSED WITH THE FOLLOWING OPTIONS:
CHECK/NO PRINT/4K/NO IMAGECOPY/NON-SEGMENT/NUMPARTS=0/NO OBIDXLAT/NO VALUE/NO RESET/ / / /
DSSIZE= /PIECESIZ= /
DSN1998I INPUT DSNAME = DSNDB1G.DSNDBC.DSNDB06.SYSDBASE.I0001.A001 , VSAM
DSN1997I OUTPUT DSNAME = NULLFILE , SEQ
DSN1991I UNCLUSTERED DATA DETECTED. RID: '0000006D03'X TABLE: SYSTABLESPACE INDEX KEY: WRKDBD1 DSN32K01
DSN1991I UNCLUSTERED DATA DETECTED. RID: '0000006D02'X TABLE: SYSTABLESPACE INDEX KEY: TMPDBD1 TEMPTS01
DSN1994I DSN1COPY COMPLETED SUCCESSFULLY, 00001250 PAGES PROCESSED

```

We submitted a reorg of DSNDB06.SYSDBASE to recluster the data; using DSN1COPY again with the CHECK option against DSNDB06.SYSDBASE then produced a return code of zero.

Before DB2 Version 8, if DBINFO was used to define external functions or procedures, the routine body must be recompiled and rebound to correctly reference ASCII or Unicode CCSID fields in DBINFO. **Migration Step 20** deals with this. The following query can be executed to identify those routines that might need to be recompiled and rebound because they reference ASCII or Unicode CCSID fields in DBINFO:

```
SELECT SCHEMA,NAME FROM SYSIBM.SYSROUTINES WHERE DBINFO='Y';
```

We had no such routines defined on our system.


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

For those views found to have view regeneration errors, we issued the following command:

```
ALTER VIEW view_name REGENERATE;
```

In **Migration Step 26** we took another image copy of the directory and catalog after they were successfully migrated to V8, and submitted job IDIRCAT8; recall that this job is located in DB2.JOBS and is a modified version of DSNTIJIC (see **Migration Step 5** for details). Execution of IDIRCAT8 completed successfully.

Optional **Migration Step 27** verifies the DB2 Version 8 subsystem that is now in Compatibility Mode. For this step, only the following Version 7 IVP jobs can be submitted:

1. Version 7 Phase 2 IVP Applications
 - a. DSNTJ2A - All steps except the first two
 - b. DSNTJ2C - Only step PH02CS04, statement RUN PROGRAM(DSN8BC3) PLAN(DSN8BH61) is to be executed
 - c. DSNTJ2D - Only step PH02DS03, statement RUN PROGRAM(DSN8BD3) PLAN(DSN8BD61) is to be executed
 - d. DSNTJ2E - Only step PH02ES04, statement RUN PROGRAM(DSN8BE3) PLAN(DSN8BE61) is to be executed
 - e. DSNTJ2F - Only step PH02FS03, statement RUN PROGRAM(DSN8BF3) PLAN(DSN8BF61) is to be executed
 - f. DSNTJ2P - Execute step PH02PS05.
2. Version 7 Phase 3 IVP Applications
 - a. ISPF-CAF applications, with the exception of DSNTJ3C and DSNTJ3P. Note that you must temporarily place the Version 7 SDSNSPFP panel library ahead of the Version 8 SDSNSPFP panel library in the ISPLIB concatenation. This permits the plans that were migrated from Version 7 to be used.

Of the IVP jobs listed, we only issued the first, DSNTJ2A, under Version 7. It is therefore the only IVP that we could use in Version 8 Compatibility Mode. For the record, if it is desired to execute the remaining IVPs as part of the verification of DB2 Version 8 Compatibility Mode, they must first be executed under Version 7 in their entirety before starting the Version 8 migration process and remain available for use after the migration to Version 8 Compatibility mode has been completed.

Before executing DSNTJ2A, we performed the following actions:

- We updated JOBLIB statements to reflect the DB2 Version 8 load library.
- We edited proc DSN8EAE1 (used by the employee sample table) and copied it from the Version 7 exit library to the Version 8 exit library.

When we completed these tasks, we ran DSNTJ2A and it produced a return code of 4 as the result of placing tables DSN8D71U.NEWDEPT and DSN8D71U.NEWPHONE in COPY PENDING status. This was as expected.

Finally, optional **Migration Step 28** 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* under topic 2.6.24, "Enabling stored procedures after installation". Since we had already enabled WLM stored procedures under DB2 Version 7, this step was skipped.

DB2 V7 and V8 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 8 compatibility mode was DBG1. 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 7 startup procs for the member being upgraded with their Version 8 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.db2810.sdsnc1st(dsntinst)'
```

We filled in the first panel as shown:

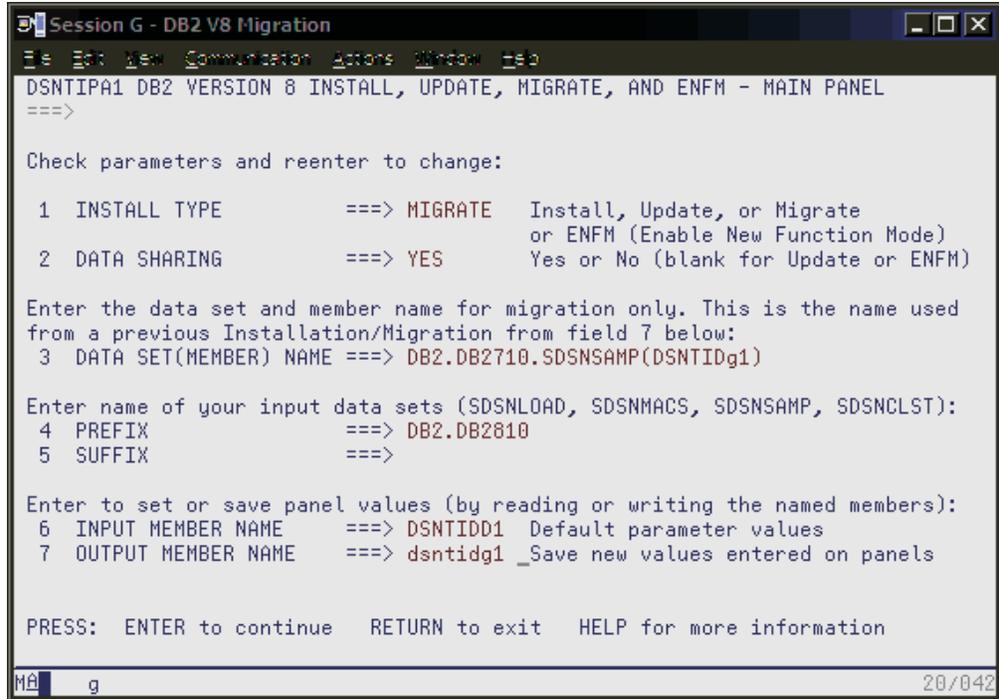


Figure 14. Executing DSNTINST in preparation for migrating the next member of the data sharing group

Pressing enter, we obtained the following pop-up screen:

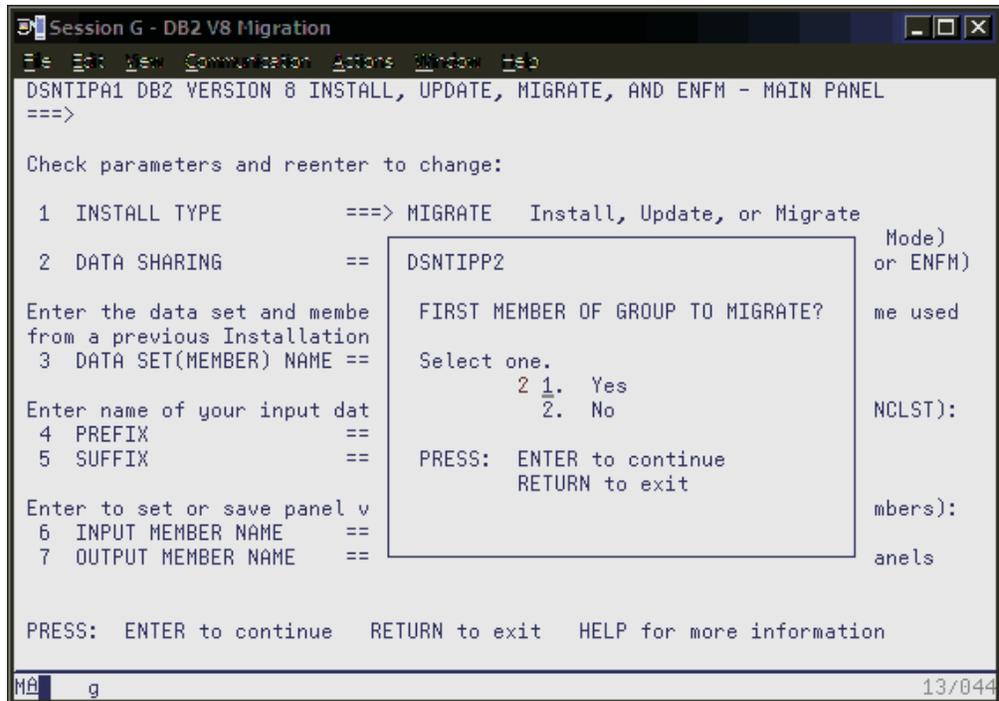


Figure 15. DSNTIPP2 pop-up screen

From this point, we scrolled 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 DBG1, as shown in Figure 16.

```

Session G - DB2 V8 Migration
File Edit View Communicator Actions Window Help
DSNTIPT          MIGRATE DB2 - DATA SET NAMES PANEL 1
===>
DSNT434I Warning,data sets marked with asterisks exist and will be overwritten
Data sets allocated by the installation CLIST for edited output:
* 1  TEMP CLIST LIBRARY  ==> DB2.DB2810.NEW.SDSNTEMP
  2  SAMPLE LIBRARY     ==> DB2.DB2810.dbg1_SDSNSAMP
Data sets allocated by the installation jobs:
* 3  CLIST LIBRARY      ==> DB2.DB2810.NEW.SDSNCLST
  4  APPLICATION DBRM   ==> DB2.DB2810.DBRMLIB.DATA
  5  APPLICATION LOAD   ==> DB2.DB2810.RUNLIB.LOAD
  6  DECLARATION LIBRARY==> DB2.DB2810.SRCLIB.DATA
Data sets allocated by SMP/E and other methods:
  7  LINK LIST LIBRARY  ==> DB2.DB2810.SDSNLINK
  8  LOAD LIBRARY       ==> DB2.DB2810.SDSNLOAD
  9  MACRO LIBRARY      ==> DB2.DB2810.SDSNMACS
 10  LOAD DISTRIBUTION  ==> DB2.DB2810.ADSNLOAD
 11  EXIT LIBRARY       ==> DB2.DB2810.SDSNEXIT
 12  DBRM LIBRARY       ==> DB2.DB2810.SDSNDBRM
 13  IRLM LOAD LIBRARY  ==> DB2.DB2810.SDXRRESL
 14  IVP DATA LIBRARY  ==> DB2.DB2810.SDSNIYPD
 15  INCLUDE LIBRARY    ==> DB2.DB2810.SDSNC.H

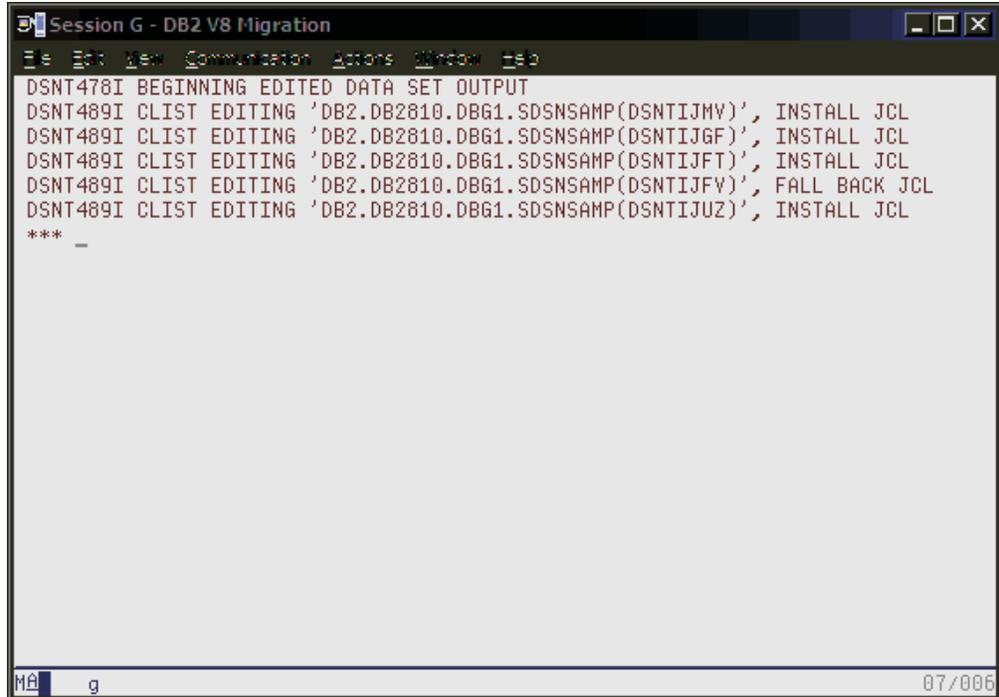
PRESS:  ENTER to continue  RETURN to exit  HELP for more information

MA  g  06/046

```

Figure 16. DSNTIPT - Data Set Names Panel 1

We placed the tailored migration JCL in DB2.DB2810.DBG1.SDSNSAMP as can be seen in Figure 17 on page 80:



```
Session G - DB2 V8 Migration
File Edit View Communicator Actions Window Help
DSNT478I BEGINNING EDITED DATA SET OUTPUT
DSNT489I CLIST EDITING 'DB2.DB2810.DBG1.SDSNSAMP(DSNTIJHY)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBG1.SDSNSAMP(DSNTIJGF)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBG1.SDSNSAMP(DSNTIJFT)', INSTALL JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBG1.SDSNSAMP(DSNTIJFY)', FALL BACK JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBG1.SDSNSAMP(DSNTIJUZ)', INSTALL JCL
***
-
Mâ g 07/006
```

Figure 17. Tailored migration JCL placed in DB2.DB2810.DBG1.SDSNSAMP

Next, we removed the SMP/E step, brought DBG1 down, and ran DSNTIJUZ. DSNTIJUZ was submitted and ran successfully.

Next, we used steps RENAME and DSNTIPM of job DSNTIJMV to rename the existing Version 7 startup procedures for DBG1 and to add the new Version 8 startup procedures to proclib.

We then started DBG1 successfully in compatibility mode, as can be seen in Figure 18 on page 81:

```

Session G - DB2 V8 Migration
-----
Display Filter View Print Options Help
-----
SDSF OUTPUT DISPLAY DBG1MSTR S0063516 DSID      2 LINE 52      COLUMNS 17- 96
COMMAND INPUT ==>          SCROLL ==> PAGE
*** BEGIN DISPLAY OF GROUP(DSNDB1G ) GROUP LEVEL(810) MODE(C)
                                PROTOCOL LEVEL(1) GROUP ATTACH NAME(DB1G)
-----
DB2 MEMBER  ID  SUBSYS  CHDPREF  STATUS  DB2 SYSTEM  IRLM  IRLMPROC
-----
DBA1      4  DBA1   @DBA1   ACTIVE  710 JA0    IRA1  DBA1IRLM
DBB1      7  DBB1   @DBB1   ACTIVE  710 JB0    IRB1  DBB1IRLM
DBC1      6  DBC1   @DBC1   ACTIVE  710 JC0    IRC1  DBC1IRLM
DBD1      5  DBD1   @DBD1   ACTIVE  810 J90    IRD1  DBD1IRLM
DBE1      3  DBE1   @DBE1   ACTIVE  710 JE0    IRE1  DBE1IRLM
DBF1      2  DBF1   @DBF1   ACTIVE  710 JF0    IRF1  DBF1IRLM
DBG1     10  DBG1   @DBG1   ACTIVE  810 J90    IRG1  DBG1IRLM
DBH1     11  DBH1   @DBH1   QUIESCED 710 JB0    IRH1  DBH1IRLM
DBI1     12  DBI1   @DBI1   ACTIVE  710 JE0    IRI1  DBI1IRLM
DBZ1      1  DBZ1   @DBZ1   ACTIVE  710 Z0     IRZ1  DBZ1IRLM
DB81      9  DB81   @DB81   ACTIVE  710 J80    IR81  DB81IRLM
DB91      8  DB91   @DB91   ACTIVE  710 J90    IR91  DB91IRLM
-----
SCA  STRUCTURE SIZE:  9216 KB, STATUS= AC,  SCA IN USE:  20 %
MA  g  04/021

```

Figure 18. DBG1 started in compatibility mode

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

```

Session G - DB2 V8 Migration
-----
Display Filter View Print Options Help
-----
SDSF OUTPUT DISPLAY DBA1MSTR S0067240 DSID      2 LINE 95      COLUMNS 17- 96
COMMAND INPUT ==>          SCROLL ==> PAGE
*** BEGIN DISPLAY OF GROUP(DSNDB1G ) GROUP LEVEL(810) MODE(C)
                                PROTOCOL LEVEL(1) GROUP ATTACH NAME(DB1G)
-----
DB2 MEMBER  ID  SUBSYS  CHDPREF  STATUS  DB2 SYSTEM  IRLM  IRLMPROC
-----
DBA1      4  DBA1   @DBA1   ACTIVE  810 JA0    IRA1  DBA1IRLM
DBB1      7  DBB1   @DBB1   ACTIVE  810 JB0    IRB1  DBB1IRLM
DBC1      6  DBC1   @DBC1   ACTIVE  810 JC0    IRC1  DBC1IRLM
DBD1      5  DBD1   @DBD1   ACTIVE  810 JA0    IRD1  DBD1IRLM
DBE1      3  DBE1   @DBE1   ACTIVE  810 JE0    IRE1  DBE1IRLM
DBF1      2  DBF1   @DBF1   ACTIVE  810 JF0    IRF1  DBF1IRLM
DBG1     10  DBG1   @DBG1   QUIESCED 810 J90    IRG1  DBG1IRLM
DBH1     11  DBH1   @DBH1   QUIESCED 810 J90    IRH1  DBH1IRLM
DBI1     12  DBI1   @DBI1   ACTIVE  810 JE0    IRI1  DBI1IRLM
DBZ1      1  DBZ1   @DBZ1   ACTIVE  810 Z0     IRZ1  DBZ1IRLM
DB81      9  DB81   @DB81   ACTIVE  810 J80    IR81  DB81IRLM
DB91      8  DB91   @DB91   ACTIVE  810 J90    IR91  DB91IRLM
-----
SCA  STRUCTURE SIZE:  9216 KB, STATUS= AC,  SCA IN USE:  20 %
MA  g  04/021

```

Figure 19. All members now in compatibility mode

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 8. The process is outlined below.

Preparing for new function mode

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

- Ensure buffer pools BP8K0, BP16K0, BP32K exist, and in a data sharing environment that their corresponding group buffer pools have been defined (GBP8K0, GBP16K0, and GBP32K).
- An image copy of the catalog and directory must be taken before executing DSNTIJNE, which converts the DB2 catalog to Unicode for the first time.
- Increase the size of shadow data sets (panel DSNTIP01 of the installation CLIST) in order to support longer object names in the DB2 catalog; depending on their current size, you might have to increase the size of the catalog table space and index space as well.
- Run the installation CLIST using the ENFM option on panel DSNTIPA1.

After attending to the first three items, the installation CLIST was executed; panel DSNTIPA1 was completed as shown in Figure 20:

```

Session G - DB2 V8 Migration
DB2 VERSION 8 INSTALL, UPDATE, MIGRATE, AND ENFM - MAIN PANEL
===>
DSNT401I Warning - help is not installed
Check parameters and reenter to change:

 1  INSTALL TYPE          ===> enfm      Install, Update, or Migrate
                                     or ENFM (Enable New Function Mode)
 2  DATA SHARING        ===>           Yes or No (blank for Update or ENFM)

Enter the data set and member name for migration only. This is the name used
from a previous Installation/Migration from field 7 below:
 3  DATA SET(MEMBER) NAME ===>

Enter name of your input data sets (SDSNLOAD, SDSNMACS, SDSNSAMP, SDSNCLST):
 4  PREFIX                ===> DB2.DB2810
 5  SUFFIX                 ===>

Enter to set or save panel values (by reading or writing the named members):
 6  INPUT MEMBER NAME     ===> dsntidd1  Default parameter values
 7  OUTPUT MEMBER NAME    ===> dsntidd1  _Save new values entered on panels

PRESS:  ENTER to continue  RETURN to exit  HELP for more information

MÁ g 20/042

```

Figure 20. Executing DSNTINST in preparation for enabling-new-function-mode

Press enter twice to display panel DSNTIP00:

```

Session G - DB2 V8 Migration
DSNTIP00  ENABLE NEW FUNCTION MODE FOR DB2 - SHADOW DATA SET ALLOCATION
===>
  OBJECT      DASD DEVICE      VOL/SERIAL      PRIMARY RECS      SECONDARY RECS
 1 SPT01      ==> 3390         ==> DB2C01       ==> 636            ==> 636
 2 SYSDBASE   ==> 3390         ==> DB2C01       ==> 7507           ==> 7507
 3 SYSDBAUT   ==> 3390         ==> DB2C01       ==> 551            ==> 551
 4 SYSDDF     ==> 3390         ==> DB2C01       ==> 165            ==> 165
 5 SYSGPAUT   ==> 3390         ==> DB2C01       ==> 552            ==> 552
 6 SYSGROUP   ==> 3390         ==> DB2C01       ==> 55             ==> 55
 7 SYSGRTNS   ==> 3390         ==> DB2C01       ==> 165            ==> 165
 8 SYSHIST    ==> 3390         ==> DB2C01       ==> 165            ==> 165
 9 SYSJAVA    ==> 3390         ==> DB2C01       ==> 165            ==> 165
10 SYSOBJ     ==> 3390         ==> DB2C01       ==> 708            ==> 708
11 SYSPKAGE   ==> 3390         ==> DB2C01       ==> 1241           ==> 1241
12 SYSPLAN    ==> 3390         ==> DB2C01       ==> 1410           ==> 1410
13 SYSSEQ     ==> 3390         ==> DB2C01       ==> 165            ==> 165
14 SYSSEQ2    ==> 3390         ==> DB2C01       ==> 165            ==> 165
15 SYSSTATS   ==> 3390         ==> DB2C01       ==> 551            ==> 551
16 SYSSTR     ==> 3390         ==> DB2C01       ==> 77             ==> 77
17 SYSUSER    ==> 3390         ==> DB2C01       ==> 488            ==> 488
18 SYSVIEWS   ==> 3390         ==> DB2C01       ==> 4029           ==> 4029
19 INDEXES    ==> 3390         ==> DB2C01       Catalog and directory index shadows
PRESS: ENTER to continue  RETURN to exit  HELP for more information
MA g 02/007

```

Figure 21. DSNTIP00 panel

We accepted calculated values and pressed enter to continue:

```

Session G - DB2 V8 Migration
DSNTIP01  ENABLE NEW FUNCTION MODE FOR DB2 - IMAGE COPY DATA SET ALLOCATIONS
===>
Enter characteristics for ENFM image copy data set allocation
1 COPY DATA SET NAME PREFIX ==> DB2.DB2810.IMAGCOPY
2 COPY DATA SET DEVICE TYPE ==> 3390
PRESS: ENTER to continue  RETURN to exit  HELP for more information
MA g 02/007

```

Figure 22. Image copy data set allocations on panel DSNTIP01

After updating the high-level-qualifier to be used for image copy data sets (DB2.IC.DB1G), we pressed enter and the following warning message appeared:

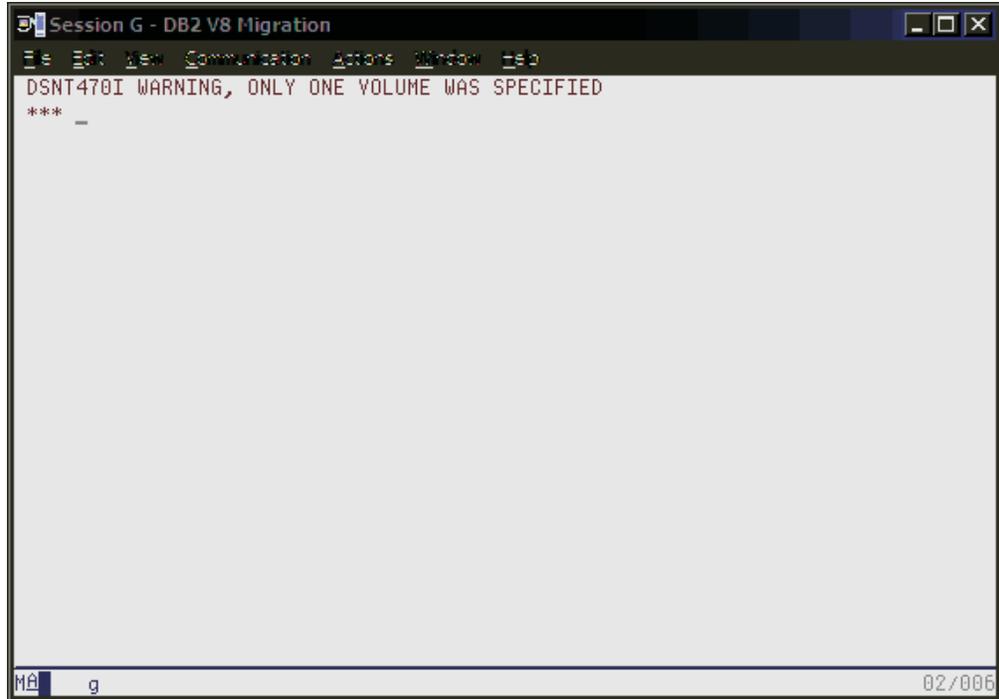


Figure 23. DSNT470I Warning message, only one volume was specified

After pressing enter, we obtained panel DSNTIP02:

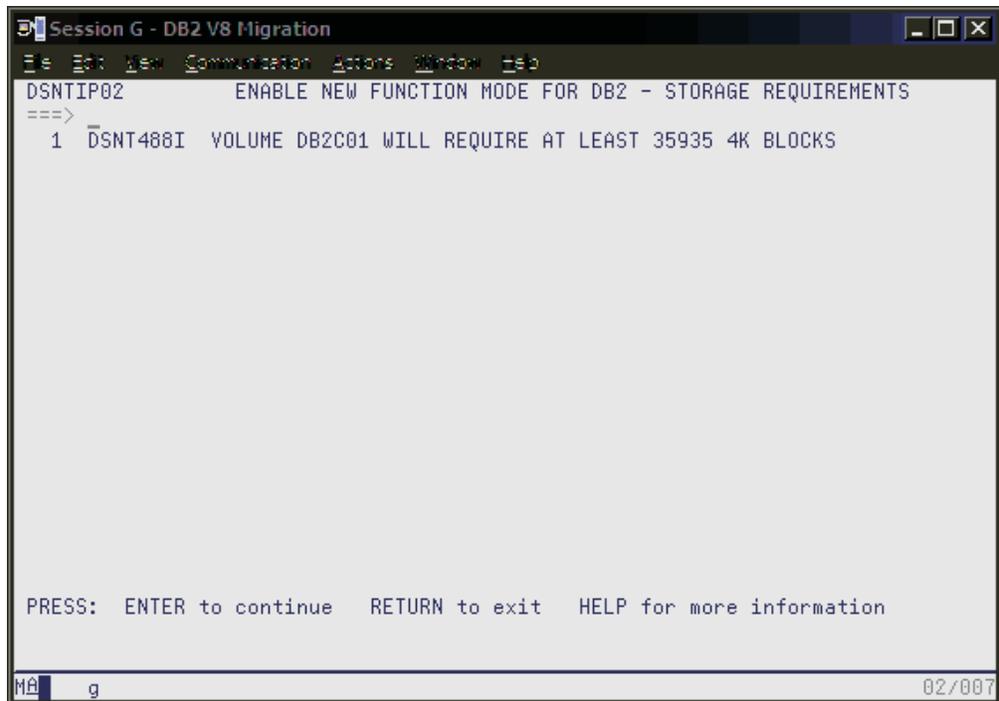


Figure 24. Message DSNT488I displayed on panel DSNTIP02

This was the last panel displayed. When we pressed enter, the generation of the enabling-new-function mode job along with the DB2 Version 8 sample jobs, occurred as shown in the following three screen images:

```

Session G - DB2 V8 Migration
File Edit New Communicator Actions Window Help
DSNT478I BEGINNING EDITED DATA SET OUTPUT
DATASET DB2.DB2810.DBD1.SDSNSAMP COMPRESSED AT 08:32:50
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJNE)', ENFM PROCESSING
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJNF)', TURN NEW FUNCTION
MODE ON
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJNH)', HALT ENFM PROCESSI
NG
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJEN)', CHANGE FROM NFM TO
ENFM
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJNR)', CONVERT RLST FOR L
ONG NAMES
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTIJMC)', SWITCH METADATA ME
THODS TO NFM
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTESC)', SAMPLE DATA
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTESD)', SAMPLE DATA
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTESA)', SAMPLE DATA
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTESE)', SAMPLE DATA
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ0)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ1)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ1L)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ1P)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ1S)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ1T)', SAMPLE JCL
***
MA g 24/006

```

Figure 25. DSNT478I beginning data set output

```

Session G - DB2 V8 Migration
File Edit New Communicator Actions Window Help
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ1U)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ2A)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ2C)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ2D)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ2E)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ2F)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ2P)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ3C)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ3P)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ3M)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ4C)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ4P)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ5A)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ5C)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ5P)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ6U)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ7)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ71)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ73)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ75)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ76)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ77)', SAMPLE JCL
DSNT489I CLIST EDITING 'DB2.DB2810.DBD1.SDSNSAMP(DSNTJ78)', SAMPLE JCL
***
MA g 24/006

```

Figure 26. DSNT489I CLIST editing

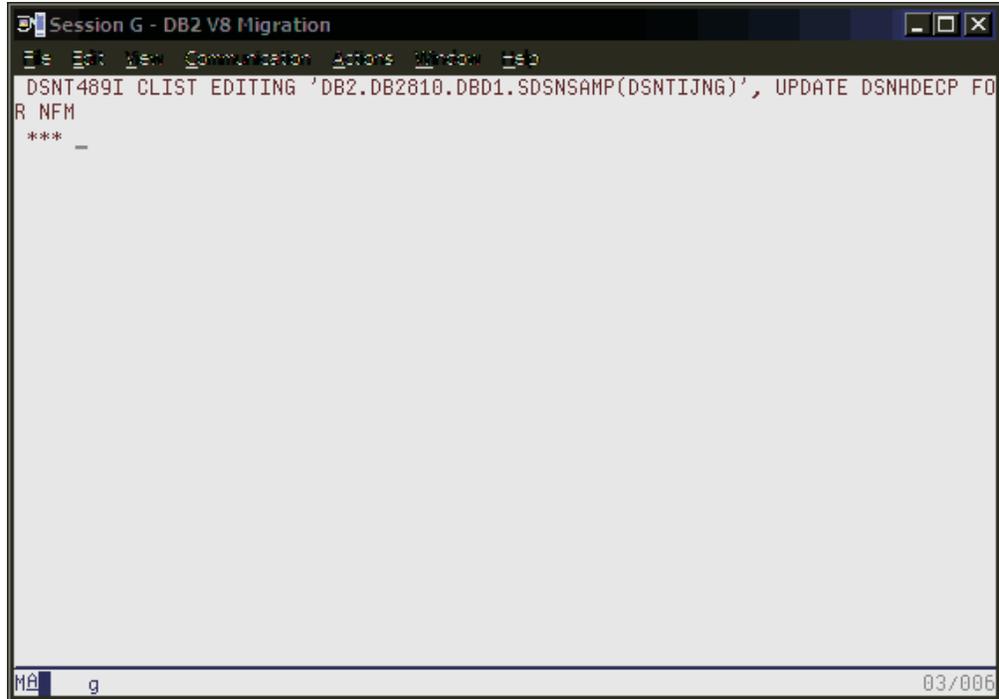


Figure 27. Screen showing completion of the preparation before enabling Version 8 new function mode

Enabling new function mode

Attention: Before proceeding, ensure that an image copy of the catalog and directory is taken.

The first step in enabling new function mode is to execute DSNTIJNE, which among other things converts the DB2 catalog to Unicode. The DISPLAY GROUP DETAIL command can be used during DSNTIJNE processing to determine how the enabling-new-function mode process is proceeding. It will display the names of the DB2 system table spaces and whether or not new function mode has been enabled yet, as shown in Table 13 on page 87:

Table 13. DB2 system table spaces and whether or not new function mode has been enabled yet.

Table Space	Enabled New Function Mode
SYSVIEWS	YES
SYSDBASE	YES
SYSDBAUT	YES
SYSDDF	NO
SYSGPAUT	NO
SYSGROUP	NO
SYSGRNTS	NO
SYSHIST	NO
SYSJAVA	NO
SYSOBJ	NO
SYSPKAGE	NO
SYSPLAN	NO
SYSSEQ	NO
SYSSEQ2	NO
SYSSTATS	NO
SYSSTR	NO
SYSUSER	NO
SPT01	NO

We submitted DSNTIJNE and then issued the DISPLAY GROUP DETAIL command throughout DSNTIJNE processing to view our progress. Note that the DISPLAY GROUP DETAIL output showed that we were now in enabling new function mode, as evidenced by MODE(E).

Upon successful completion of DSNTIJNE, we took another image copy of the catalog and directory.

Next, we executed job DSNTIJNF, which is used to verify that the conversion of the DB2 catalog and directory. It completed successfully and produced a return code of zero. A DISPLAY GROUP DETAIL at this point reveals that we are now in new function mode (note MODE(N) in Figure 28 on page 88):

```

Session G - DB2 V8 Migration
-----
ISFPCU41  CONSOLE JCORRY                                LINE 23 RESPONSES NOT SHOWN
COMMAND INPUT ==> -                                     SCROLL ==> PAGE
RESPONSE=J90
DSN7100I  @DBD1 DSN7GCHD
*** BEGIN DISPLAY OF GROUP(DSNDB1G ) GROUP LEVEL(810) MODE(N)
          PROTOCOL LEVEL(1)  GROUP ATTACH NAME(DB1G)
-----
DB2      DB2      DB2      DB2      DB2      DB2      DB2      DB2      DB2      DB2
MEMBER   ID       SUBSYS  CMDPREF  STATUS   LVL NAME  IRLM     IRLM     IRLM     IRLM
-----
DBA1     4   DBA1   @DBA1   QUIESCED 810 JAO     IRA1     DBA1IRLM
DBB1     7   DBB1   @DBB1   QUIESCED 810 JBO     IRB1     DBB1IRLM
DBC1     6   DBC1   @DBC1   QUIESCED 810 JCO     IRC1     DBC1IRLM
DBD1     5   DBD1   @DBD1   ACTIVE   810 J90     IRD1     DBD1IRLM
DBE1     3   DBE1   @DBE1   QUIESCED 810 JEO     IRE1     DBE1IRLM
DBF1     2   DBF1   @DBF1   QUIESCED 810 JFO     IRF1     DBF1IRLM
DBG1    10   DBG1   @DBG1   QUIESCED 810 J90     IRG1     DBG1IRLM
DBH1    11   DBH1   @DBH1   QUIESCED 810 J90     IRH1     DBH1IRLM
DBI1    12   DBI1   @DBI1   QUIESCED 810 JEO     IRI1     DBI1IRLM
DBZ1     1   DBZ1   @DBZ1   QUIESCED 810 Z0     IRZ1     DBZ1IRLM
DB81     9   DB81   @DB81   QUIESCED 810 J80     IR81     DB81IRLM
DB91     8   DB91   @DB91   QUIESCED 810 J90     IR91     DB91IRLM
-----

```

Figure 28. The DISPLAY GROUP command shows the data sharing group is now in new function mode

Running in new function mode

When we were in new function mode, we needed to run DSNTIJNG; it modifies DSNHDECP and allows new-function SQL statements introduced with Version 8 to be accepted by the DB2 precompiler by default. Note that in a data sharing environment where multiple DSNHDECP modules are in use, the jobs used to maintain the DSNHDECP modules must be updated to specify NEWFUN=YES. In our environment, only a single DSNHDECP module exists, therefore we ran DSNTIJNG successfully. (Note that we did not execute the last step, which deals with SMP/E.)

During the process of enabling-new-function mode, DATA CAPTURE was set to NONE on all of the DB2 catalog tables with the exception of SYSCOPY. Now that we are in new function mode, DATA CAPTURE must be re-enabled. This is a manual process and is performed by issuing the following ALTER command:

```
ALTER TABLE SYSIBM.SYSTABLES DATA CAPTURE CHANGES;
```

Also during enabling-new-function mode processing, two DB2 catalog tables that are not required by Version 8 were deleted. Once in new function mode, the corresponding VSAM data set for the index DSNKCX01 can be manually deleted.

An optional step can be executed once in new function mode that can be performed to convert SYSIBM.DSNRLST01, (the Resource Limit Specification Table, or RLST), to provide long name support for the authorization id (AUTHID), collection id (RLFCOLLN), and package id (RLFPKG) columns. Note that this step is only required if you would like to convert the Version 7 RLST or some other RLST (by

modifying the job) to support long names. We decided to enable long name support for the aforementioned columns of RLST and executed DSNTIJNR successfully.

Note that it is possible to create a larger BSDS once in new function mode, thereby providing support for up to 93 active log data sets and 10,000 archive log data sets per copy. Our BSDS was of a sufficient size for the time being, so we did not run the DSNJCNVB conversion utility as outlined in the *DB2 Installation Guide*.

Finally, 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.

The following DSNDB06 indexes were placed in informational image copy status:

```
DSNDDX02
DSNDPX01
DSNDRX01
DSNDSX01
DSNDTX01
DSNDXX03
DSNPPH01
```

We image copied these indexes.

We also placed DSNRLST objects DSNRLS01 and DSNARL01 in advisory reorg status. Therefore, we reorganized table space DSNRLST.DSNRLS01.

Verifying the installation using the sample applications

Using the sample applications provided in DB2.DB2810.DBD1.SDSNSAMP, we performed verification of DBD1's migration to DB2 Version 8 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, DSNTJ0, 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.DB2810.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.DB2810.PROCLIB. This library must be APF authorized (we dynamically added it to the APF authorization list before proceeding).

Beginning with **Phase 1**, which is used to create and load sample tables, we ran job DSNTEJ1. It produced the return codes as shown in Table 62 in the *DB2 Installation Guide*.

We prepared DSNTEP2 for DB2 Version 8 using job DSNTEJ1L, which produced a return code of zero. Job DSNTEJ1P was not executed as customization of DSNTEP2 was not required.

Job DSNTEJ1U installs DB2 Unicode support, but should only be run if the operating system is capable of supporting Version 7 Unicode. Referring to the program directory, Unicode support requires OS/390 Version 2 Release 9 or higher. Since our system is on z/OS Version 1 Release 6, we were able to execute DSNTEJ1U successfully.

Phase 2 tests the batch environment. Job DSNTEJ2A was executed to prepare assembler program DSNTIAUL. All steps produced the expected return codes.

Problems encountered:

When we ran job DSNTEJ2C next we ran into some problems. When we first ran this job, we received the following error:

```
IGY0S4003-E  INVALID OPTION PGMNAME(LONGUPPER) WAS FOUND AND DISCARDED
```

To correct this, we modified the parameters on the EXEC statement of steps PH02CS02 and PH02CS03 in job DSNTEJ2C. Originally, they were:

```
//PH02CS02 EXEC DSNHICOB, MEM=DSN8MCG,
//          COND=(4,LT),
//          PARM.PC=('HOST (IBMCOB)', APOST, APOSTSQL, SOURCE,
//          NOXREF, 'SQL (DB2)', 'DEC(31)'),
//          PARM.COB=(NOSEQUENCE, QUOTE, RENT, 'PGMNAME(LONGUPPER)'),
//          PARM.LKED='LIST, XREF, MAP, RENT'
.
.
.
//PH02CS03 EXEC DSNHICOB, MEM=DSN8BC3,
//          COND=(4,LT),
//          PARM.PC=('HOST (IBMCOB)', APOST, APOSTSQL, SOURCE,
//          NOXREF, 'SQL (DB2)', 'DEC(31)'),
//          PARM.COB=(NOSEQUENCE, QUOTE, RENT, 'PGMNAME(LONGUPPER)')
```

We updated them as shown below:

```
//PH02CS02 EXEC DSNHICOB, MEM=DSN8MCG,
//          COND=(4,LT),
//          PARM.PC=('HOST(IBMCOB)', APOST, APOSTSQL, SOURCE,
//          NOXREF, 'SQL(DB2)', 'DEC(31)'),
//          PARM.COB=(NOSEQUENCE, APOST, RENT),
//          PARM.LKED='LIST, XREF, MAP, RENT'
.
.
.
//PH02CS03 EXEC DSNHICOB, MEM=DSN8BC3,
//          COND=(4,LT),
//          PARM.PC=('HOST(IBMCOB)', APOST, APOSTSQL, SOURCE,
//          NOXREF, 'SQL(DB2)', 'DEC(31)'),
//          PARM.COB=(NOSEQUENCE, APOST, RENT)
```

Our modifications are highlighted above. Note that parameter PGMNAME(LONGUPPER) was removed altogether. After making these modifications, the job ran successfully and generated the expected return codes.

Because we do not use C, C++, or Fortran to access DB2, installation verification programs DSNTEJ2D, DSNTEJ2E and DSNTEJ2F were not executed.

We ran job DSNTEJ2P next to test PL/I program preparation procedures. It ran successfully.

Because we do not have access to C++ and did not complete the fields pertaining to C++ on the installation panel DSNTIPU, job DSNTEJ2U was not generated by the installation CLIST. This completed **Phase 2** of the installation verification procedures.

Phase 3 tests SPUFI, DRDA[®] access, dynamic SQL, and TSO connections to DB2 Version 8.

To test SPUFI, we used members DSNTESA, DSNTESC and DSNTESI of data set DB2.DB2810.DBD1.SDSNSAMP as input to SPUFI. We ran all these members successfully with the exception of DSNTESC, which failed on each of the insert statements. The DB2 Version 7 tables DSN8710.PLAN_TABLE, DSN8710.DSN_FUNCTION_TABLE, and DSN8710.DSN_STATEMNT_TABLE did not exist, causing the corresponding subselect to fail. When we commented out the insert statements, DSNTESC completed successfully.

We skipped running SPUFI at remote non-DB2 systems as none were available, and moved on to installation of the ISPF/CAF sample application. We removed the DSNHICOB parameter PGMNAME(LONGUPPER), and changed the QUOTE parameter to APOST before executing DSNTEJ3C. Next, we ran DSNTEJ3C successfully and generated the expected return codes. Finally, we ran DSNTEJ3P successfully and produced the designated return codes.

Chapter 6. Migrating to IMS Version 9

We migrated our IMS systems from IMS Version 8 Release 1 to IMS Version 9 Release 1. We completed the migration of our IMS systems in the following stages:

1. We migrated one of our IMS systems from IMS V8 to V9 on an IBM @server zSeries 990 server.
2. We migrated another IMS system to V9 on an IBM @server zSeries 800 server.
3. We migrated another IMS system to V9 on an IBM @serverzSeries 900 server.
4. We migrated all remaining IMSs to Version 9 Release 1, except one, which needs to remain at Version 8 Release 1 for testing that is still in progress.

As soon as our IMS Version 8 Release 1 testing is completed, we will migrate the final IMS to Version 9 Release 1. We have been running with a mixed release IMSplex for approximately five months.

We used information from the following IMS V9.1 publications to perform the migration:

- IMS Version 9: Release Planning Guide, GC17-7831
- IMS Version 9: Installation Volume 1: Installation Verification , GC18-7822
- IMS Version 9: Installation Volume 2: System Definition and Tailoring, GC18-7823

We successfully performed the migration according to the instructions in the product publications. The following sections describe some of our experiences and observations.

Installing availability enhancements: IMS Version 9 provides two major enhancements for availability.

1. **z/OS Resource Manager Services:** IMS dynamically installs its Resource Manager cleanup routine; you do not need to install the DFSMRCL0 module as part of the IMS installation. Registration of the IMS Resource Manager cleanup routine with the operating system is done automatically during IMS startup.

The Resource Manager is registered dynamically only for IMS Version 9 or later. If you use earlier IMS releases, or use both IMS Version 9 and earlier IMS releases, you must still install the DFSMRCL0 module as part of the IMS installation. The DFSMRCL0 module must be the highest version prior to IMS Version 9. When all IMSs (control region and batch regions) are IMS Version 9 or later, you can remove DFSMRCL0 from SYS1.LPALIB and remove the IEAVTRML CSECT of z/OS module IGC0001C. You must remove the name DFSMRCL0 from the IEAVTRML CSECT before you remove the module from SYS1.LPALIB. If the name is still in IEAVTRML but the module is not in SYS1.LPALIB, z/OS IPL will fail.

Because we are running with both V8 and V9, we installed the V9 level of the DFSMRCL0 module. When we are running entirely on V9 we will uninstall the DFSMRCL0 module.

2. **DBRC Type-4 SVC Dynamic Install:** The Dynamic SVC (DFSUSVC0) utility dynamically updates the DBRC type-4 SVC module. Thus, you can apply maintenance to the DBRC type-4 SVC module without having to restart z/OS after each update. Before IMS Version 9, only the IMS type-2 SVC module could be dynamically updated. Any updates to the DBRC type-4 SVC required restarting z/OS.

You must define the IMS type-2 SVC and the DBRC type-4 SVC to z/OS before IMS starts. After you add the SVC definitions to IMS and restart z/OS, you can use the DFSUSVC0 utility to update the SVC routines without having to restart z/OS.

For more info see IMS Version 9: Release Planning Guide, GC17-7831

Using the same SVC numbers for different releases of IMS: IMS uses Type 2 supervisor calls (SVCs) in the range of 200 through 255 for batch, DBCTL, DCCTL, and DB/DC IMS control program functions, and a type 4 SVC in the same range for DBRC functions.

Note that, as in the past, the SVCs themselves are also downward compatible. For us, this means that during our migration we can use the V9 SVCs for both our IMS V9 and V8 systems.

Using the enhanced CHANGE.RECON command to upgrade the RECON data set without bringing systems down: Before migrating a system to IMS V9, we had to upgrade the RECON data set to the V9 level—you cannot migrate IMS until you do so. We used the enhanced CHANGE.RECON command, which ships as part of the V9 coexistence support SPE, to convert the RECON data set to the V9 level without shutting down active systems. This command must be run in batch mode because the RECON data set being upgraded to V9 still resides on a V8 system.

We performed the following steps to upgrade our RECON data set using the enhanced CHANGE.RECON command:

1. Installed the V9 coexistence support SPE PQ72840 and UQ82290.
2. Used the DBRC command utility, DSPURX00, to issue the CHANGE.RECON UPGRADE command. This command upgrades the RECON data set to the V9 level without shutting down all IMS activity. It also uses the DBRC I/O recovery algorithms to recover from failures during the upgrade.

The following is the output from our DSPURX00 command utility job that issued the CHANGE.RECON UPGRADE command:

```

      IMS VERSION 9 RELEASE 1  DATA BASE RECOVERY CONTROL
      CHANGE.RECON UPGRADE
      DSP0251I  RECON COPY 1 UPGRADE IS BEGINNING
      DSP0252I  RECON COPY 1 UPGRADED SUCCESSFULLY
      DSP0251I  RECON COPY 2 UPGRADE IS BEGINNING
      DSP0252I  RECON COPY 2 UPGRADED SUCCESSFULLY
      DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
      DSP0220I  COMMAND COMPLETION TIME 05.041 13:17:02.3
      IMS VERSION 9 RELEASE 1  DATA BASE RECOVERY CONTROL
      DSP0211I  COMMAND PROCESSING COMPLETE
      DSP0211I  HIGHEST CONDITION CODE = 00

```

Using a new JCL execution member for OLDS for coexistence: While we were running in coexistence mode with both IMS V9 and V8 systems, we needed two skeletal JCL execution members that kick off the online log data set (OLDS) archive—one for each release. The default skeletal JCL member for the log archive utility, ARCHJCL, points to the IMS V9 target library, RESLIB. For our V9 IMS systems, we created a new skeletal JCL execution member, ARCHJCL9, that points to the IMS V9 target library, SDFSRESL. To ensure that we invoke the new ARCHJCL9 execution member for our V9 systems, we pointed our VSPEC parameter to a new DFSVSMxx member of IMS.PROCLIB, which contains the following ARCHDEF statement:

ARCHDEF ALL MAXOLDS(1) MEMBER(ARCHJCL9)

Updating our change accumulation utilities, image copy utilities, and recovery jobs: We updated our change accumulation utilities, image copy utilities, and recovery jobs to point to the IMS V9 libraries. This is necessary for IMS V9 and V8 to coexist. After we completed the updates, all of these utilities and jobs ran successfully.

Updating IPCS and ISPF panels: We updated our IPCS and ISPF dialog panels to point to the IMS V9 libraries.

Upgrading the IMS V9 utilities: We needed to upgrade the following IMS V9 utilities:

- IMS Library Integrity Utilities for z/OS, V1.1 5655-I42 replaces our current IMS LibraryManagement Utilities 5655-E04.
- IMS Database Control Suite for z/OS, V3.1 5655-L08 replaces our current V2.2 level.
- IMS High Performance Pointer Checker for z/OS, V2.1 5655-K53 replaces our current V1.1 level.

Applying additional service for IMS V9: We did not need to apply any additional service to migrate to IMS V9.

Exploiting new functions in IMS V9: We have completed our migration to IMS V9 and we are currently evaluating ways to exploit new functions, such as High Availability Large Database (HALDB) Online Reorganization. As we implement new functions, we will report on our experiences with them in upcoming editions of our test report.

After our migration to IMS V9 is completed successfully, we will do the following:

- **Update the RECON MINVERS parm:** Use the CHANGE.RECON command to update options in the RECON status record to specify V9R1:
change.recon minivers(91)

For more information see IMS Database Recovery Control (DBRC) Guide and Reference (SC18-7818-00)

- **Uninstall DFSMRCL0:** Our final V9 migration step will be to uninstall module DFSMRCL0 from SYS1.LPALIB. See 1 on page 93 for more information.

Migrating to the integrated IMS Connect

IMS Connect was always a separately orderable product. IMS Version 9 provides an integrated IMS Connect function that offers a functional replacement for the IMS Connect tool (program number 5655-K52). The integrated IMS Connect is included in the IMS System Services function modification identifier (FMID), HMK9900; the integrated IMS Connector for Java for z/OS is included with the IMS Java FMID, JMK9906; and the Integrated IMS Connector for Java distributed can be downloaded from the IMS Web site:

www.ibm.com/software/data/ims/

For more information see IMS Version 9: Release Planning Guide, GC17-7831

During the IMS V9 migration we chose to migrate from IMS Connect V2.1 to IMS V9 integrated IMS Connect.

Migrating to IRLM Version 2 Release 2

During the IMS migration we also chose to migrate from IRLM Version 2 Release 1 to Version 2 Release 2. This was a simple migration that included the following steps:

- Updated PET.PROCLIB(IRLM) to remove unused parameters:
PC=YES
MAXCSA=12
- Updated PET.PROCLIB(IRLM) to add new IRLM 2.2 parameter:
MEMLIMIT=2G

Although we chose to do the following migrations within a very short period of time, no major problems were discovered:

- IMS V8 to V9,
- IMS Connect 2.1 to the integrated IMS Connect
- IRLM 2.1 to 2.2

This was by far, the smoothest IMS migration we have performed.

Chapter 7. Implementing the IMS Common Service Layer and the Single Point of Control

The IMS Common Service Layer (CSL) is a collection of IMS manager address spaces that provide the necessary infrastructure for systems management tasks. The IMS CSL reduces the complexity of managing multiple IMS systems by providing you with a single-image perspective in an IMSplex. That is, you can now manage multiple IMS subsystems in an IMSplex as if they were one system.

An IMS single point of control (SPOC) is a program with which you can manage operations of all IMS systems within an IMSplex.

We used the following documentation to help us implement the CSL and SPOC in our production IMSplex:

- *IMS Version 8: Common Service Layer Guide and Reference*, SC27-1293
- *IMS Version 8: Common Queue Server Guide and Reference*, SC27-1292
- *IMS Version 8: Installation Volume 2: System Definition and Tailoring*, GC27-1298

Setting up the Common Service Layer

The CSL address spaces, or CSL managers, include the operations manager (OM), resource manager (RM), and structured call interface (SCI). The CSL managers perform the following functions:

- **Operations manager (OM):** Helps control the operations of all IMS systems in an IMSplex. The OM receives processing control when an OM request (an IMS command, for example) is received by the OM application programming interface (API). All commands and responses to those commands must come through the OM API.
- **Resource manager (RM):** Helps manage resources that are shared by multiple IMS systems in an IMSplex. The RM provides the infrastructure for managing global resource information and coordinating IMSplex-wide processes.
- **Structured call interface (SCI):** Allows IMSplex members to communicate with one another. Communication between IMSplex members can occur within a single z/OS image or among multiple images. The individual IMSplex members do not need to know where the other members reside or what communication interface to use.

See Figure 29 on page 101 for a depiction of these address spaces.

Steps for setting up the CSL

We performed the following steps to set up the CSL on our z/OS production systems:

1. Added PGMNAME(BPEINI00) to the PPT (SCHED00):

```
PPT PGMNAME(BPEINI00)      /* PROGRAM NAME = BPEINI00          */
                                CANCEL /* PROGRAM CAN BE CANCELED          */
                                KEY(7) /* PROTECT KEY ASSIGNED IS 7       */
                                NOSWAP /* PROGRAM IS NON-SWAPPABLE        */
                                NOPRIV /* PROGRAM IS NOT PRIVILEGED       */
                                DSI    /* REQUIRES DATA SET INTEGRITY    */
                                PASS   /* CANNOT BYPASS PASSWORD PROTECTION */
                                SYST   /* PROGRAM IS A SYSTEM TASK        */
                                AFF(NONE) /* NO CPU AFFINITY                 */
                                NOPREF /* NO PREFERRED STORAGE FRAMES    */
```

IMS Common Service Layer and the Single Point of Control

Note: BPEINI00 can also be used to start CQS, so the CQSINIT0 entry is no longer needed. Once we migrated all of our production systems to IMS V8.1, we removed the entry for CQSINIT0.

2. Added the IMSPLEX parameter to the CQSIPxxx member on each system:

```
CQSGROUP=SQGRP,  
IMSPLEX(NAME=PROD),  
SSN=CQSA,  
STRDEFG=ALL,  
STRDEFL=PEA
```

3. Added the RSRCSTRUCTURE parameter to the CQSSGALL member:

```
STRUCTURE(  
  STRNAME=FFMSGQ_STR,  
  OVFLWSTR=FFOVFLO_STR,  
  STRMIN=0,  
  SRDSDSN1=CQS.FF.SRDS1,  
  SRDSDSN2=CQS.FF.SRDS2,  
  LOGNAME=CQS.FF.LOGSTRM,  
  OBJAVGSZ=1024,  
  OVFLWMAX=50  
)  
STRUCTURE(  
  STRNAME=FPMSGQ_STR,  
  OVFLWSTR=FPOVFLO_STR,  
  STRMIN=0,  
  SRDSDSN1=CQS.FP.SRDS1,  
  SRDSDSN2=CQS.FP.SRDS2,  
  LOGNAME=CQS.FP.LOGSTRM,  
  OBJAVGSZ=512,  
  OVFLWMAX=50  
)  
RSRCSTRUCTURE(STRNAME=CSLRMGR_PROD)
```

Note: A resource structure is not needed if only one RM is used in the IMSplex. For availability reasons, we chose to start two RMs and defined a resource structure.

4. Created a DFSCGxxx member:

```
CMDSEC=N,  
IMSPLEX=PROD,  
LEOPT=N,  
NORSCCC=(),  
OLC=LOCAL
```

5. Added the CSLG parameter to the DFSPBxxx member on each system:

```
DLINM=DLIGRP81,DBRCNM=DBRCGP81,  
AUTO=N,  
GRSNAME=IMSPETGR,  
SUF=1,  
CRC=#,LHTS=512,NHTS=512,UHTS=512,  
CMDMCS=Y,  
CSLG=PET,  
APPC=N,AOIS=S,  
FIX=HP,VSPEC=81,PRLD=DB,SPM=02,  
RSRMBR=,GRNAME=NATIVE2,  
...
```

6. Created the initialization proclib members for the CSL manager address spaces:

Example: The following is the SCI initialization member:

```

-----*
* Sample SCI Initialization Proclib Member.                                *
-----*

ARMRST=N,                                /* ARM should restart OM on failure */
SCINAME=SCI1,                             /* SCI Name (SCIID = SCI1SCI)      */
IMSPLEX(NAME=PROD)                       /* IMSplex Name                    */

```

Example: The following is the OM initialization member:

```

-----*
* Sample OM Initialization Proclib Member.                                *
-----*

ARMRST=N,                                /* ARM should restart OM on failure */
CMDLANG=ENU,                             /* Use English for Command Desc    */
CMDSEC=N,                                 /* No Command Security             */
CMDTEXTDSN=IMS810.SDFSDATA,              /*                                */
OMNAME=OM1,                               /* OM Name (OMID = OM1OM)         */
IMSPLEX(NAME=PROD)                       /* IMSplex Name (CSLPLEX1)        */

```

Example: The following is the RM initialization member:

```

-----*
* Sample RM Initialization Proclib Member.                                *
-----*

ARMRST=N,                                /* ARM should restart RM on failure */
CQSSN=CQSC,                              /*                                */
IMSPLEX(NAME=PROD,                       /* IMSPLEX NAME                    */
RSRCSTRUCTURE(STRNAME=CSLRMGR_PROD)),
RMNAME=RMC                               /* RM Name (RMID = RM1RM)         */

```

7. Created the CSL startup procedures:

Example: The following is our SCI startup procedure, CSLSCI:

```

//CSLSCI  PROC RGN=3000K,SOUT=A,
//          IMSVAR=&IMSVAR,
//          BPECFG=BPECFG00,
//          SCIINIT=000,
//          PARM1=(SCINAME=&SCINAME)
//*
//SCIPROC EXEC PGM=BPEINI00,REGION=&RGN,
// PARM='BPECFG=&BPECFG,BPEINIT=CSLSINI0,SCIINIT=&SCIINIT,&PARM1'
//*
//STEPLIB DD DISP=SHR,DSN=IMS810.&IMSVAR..SDFSRESL
//          DD DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB DD DSN=D10.PETDSW4.PROCLIB,DISP=SHR
//SYSPRINT DD SYSOUT=&SOUT
//SYSUDUMP DD SYSOUT=&SOUT
//*

```

Example: The following is our OM startup procedure, CSLOM:

```

//CSLOM  PROC RGN=3000K,SOUT=A,
//          IMSVAR=&IMSVAR,
//          BPECFG=BPECFG00,
//          OMINIT=000,
//          PARM1=(OMNAME=&OMNAME)
//*
//OMPROC EXEC PGM=BPEINI00,REGION=&RGN,
// PARM='BPECFG=&BPECFG,BPEINIT=CSLOINI0,OMINIT=&OMINIT,&PARM1'

```

IMS Common Service Layer and the Single Point of Control

```
//*
//STEPLIB DD DSN=IMS810.&IMSVAR..SDFSRESL,DISP=SHR
//          DD DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB DD DSN=D10.PETDSW4.PROCLIB,DISP=SHR
//SYSPRINT DD SYSOUT=&SOUT
//SYSUDUMP DD SYSOUT=&SOUT
//*
```

Example: The following is our RM startup procedure, CSLRM:

```
//CSLRM PROC RGN=0M,SOUT=A,
//          IMSVAR=&IMSVAR,
//          BPECFG=BPECFG00,
//          INIT=CSLRINI0,
//          RMINIT=&RMINIT
//*
//RMPROC EXEC PGM=BPEINI00,REGION=&RGN,
// PARM='BPEINIT=&INIT,BPECFG=&BPECFG,RMINIT=&RMINIT'
//STEPLIB DD DSN=IMS810.&IMSVAR..SDFSRESL,DISP=SHR
//          DD DSN=SYS1.CSSLIB,DISP=SHR
//PROCLIB DD DSN=D10.PETDSW4.PROCLIB,DISP=SHR
//SYSPRINT DD SYSOUT=&SOUT
//SYSUDUMP DD SYSOUT=&SOUT
//*
```

-
8. Updated the SCI registration exit routine (DSPSCIX0) and placed it into an authorized library:

```
PTBLEYEC DS 0H                TABLE EYECATCHER
DC C'PLEXTABL'
PLEXTABL DS 0H
DC CL(DSNL)'RECON1.PROD'      RECON NAME
DC CL(PNL)'PROD '            IMSplex name
DC XL(RCL)'00000000'         RC00 = use the IMSplex name
DC CL(DSNL)'RECON2.PROD'      RECON NAME
DC CL(PNL)'PROD '            IMSplex name
DC XL(RCL)'00000000'         RC00 = use the IMSplex name
DC CL(DSNL)'RECON3.PROD'      RECON NAME
DC CL(PNL)'PROD '            IMSplex name
DC XL(RCL)'00000000'         RC00 = use the IMSplex name
```

-
9. Defined the resource manager coupling facility structure in the CFRM policy:

```
STRUCTURE NAME(CSLRMGR_PROD)
SIZE(32000)
INITSIZE(20000)
ALLOWAUTOALT(YES)
FULLTHRESHOLD(60)
DUPLEX(ALLOWED)
PREFLIST(CF2,CF1,CF3)
```

Our CSL and SPOC configuration

Figure 29 on page 101 illustrates our the CSL and SPOC configuration in our IMSplex:

IMS Common Service Layer and the Single Point of Control

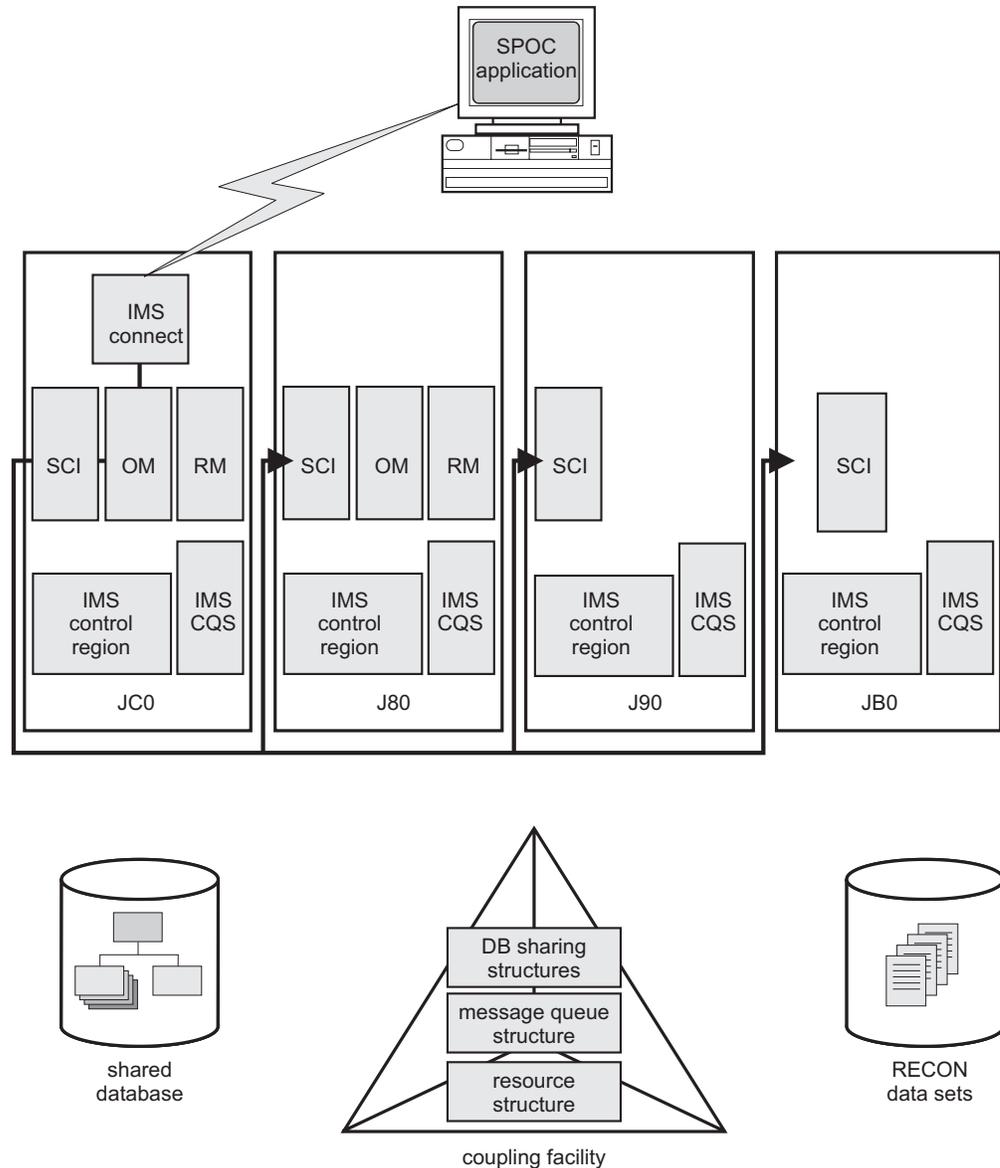


Figure 29. Our IMS CSL and SPOC configuration

Note the following about our configuration:

- We run one SCI address space on each z/OS system where IMS subsystems run.
- We only run OM and RM address spaces on systems J80 and JC0.
- We use automations to start the CSL address spaces following system IPLs. However, we found that if the RM doesn't detect the required CQS address space within 10 minutes, it terminates with a U0010-00000508 user abend. To avoid this, we added the CQS startup to automations, instead of allowing IMS to automatically start the CQS address space. This ensures that the CQS address space is available when the RM expects it.

IMS performance considerations for CSL

For the CSL address spaces, IBM recommends using the SYSSTC service class or a service class with higher importance (that is, a lower-numbered value) than the CNTL and CQS address spaces and all dependent regions. Since WLM provides

IMS Common Service Layer and the Single Point of Control

five levels of importance, a general guideline would be to group resources into service classes with the following relative importance:

IMPORTANCE		
Level	Value	Address spaces
higher	N	IRLM
	N+1	VTAM, APPC, DBRC, SCI, OM, RM
	N+2	CNTL, CQS
	N+3	DLIS
lower	N+4	dependent regions

Thus, when CPU resources are constrained, the following rules would apply:

- All dependent regions should have the lowest dispatching priority among the other IMS address spaces.
- CNTL and CQS, the address spaces with the next largest CPU consumption, should have a lower dispatching priority than the CSL address spaces.

Setting up the single point of control

A SPOC communicates with one OM address space; the OM then communicates with all of the other IMS address spaces in the IMSplex, through the SCI, as required for operations.

Steps for setting up the single point of control

We performed the following steps to set up the single point of control:

1. Verified that IMS service PQ69527 (PTF UQ73719) is installed.

2. Verified that IMS Connect is installed.
We are currently running IMS Connect V2.1. However, note that if you are running IMS Connect V1.2, you must install PQ62379 (PTF UQ69902) and PQ70216 (PTF UQ74285).

3. Updated the HWS configuration member to add the EXIT and IMSPLEX parameters:
HWS
(ID=HWSC,RACF=N)
TCPIP
(HOSTNAME=TCPIP,RACFID=RACFID,PORTID=(9999,9998,9997,9996,9995,9994,
9993,9992,9991,9990),
EXIT=(HWCSL00,HWCSL01),
MAXSOC=51,TIMEOUT=9999)
DATASTORE
(ID=IMSC,GROUP=NATIVE2,MEMBER=HWSC,TMEMBER=IMSPETJC)
IMSPLEX
(MEMBER=IMSPLEXC,TEMBER=PROD)

4. Installed DB2 UDB V8.1 (DB2 Control Center) on the workstation.
We are currently running at the FixPak 4 service level. You can get the latest FixPaks from the DB2 Technical Support Web site at www.ibm.com/cgi-bin/db2www/data/db2/udb/winos2unix/support/index.d2w/report.

Steps for setting up DB2 Control Center for the IMS SPOC

We performed the following steps to set up DB2 Control Center for the IMS SPOC:

1. Started the Control Center, then clicked **Selected** → **Add** from the menu bar.
-
2. In the **Add System** dialog box:
 - a. Selected the **IMS** button
 - b. In the **System name** box, typed the name of our IMSplex: **PROD**
 - c. In the **Host name** box, typed the IP address of the system where the IMS Connect subsystem is located
 - d. In the **Port number** box, typed the IMS Connect port number we wanted to use: **9995**
 - e. Clicked **OK**

Example: Figure 30 is an example of the **Add System** dialog box:

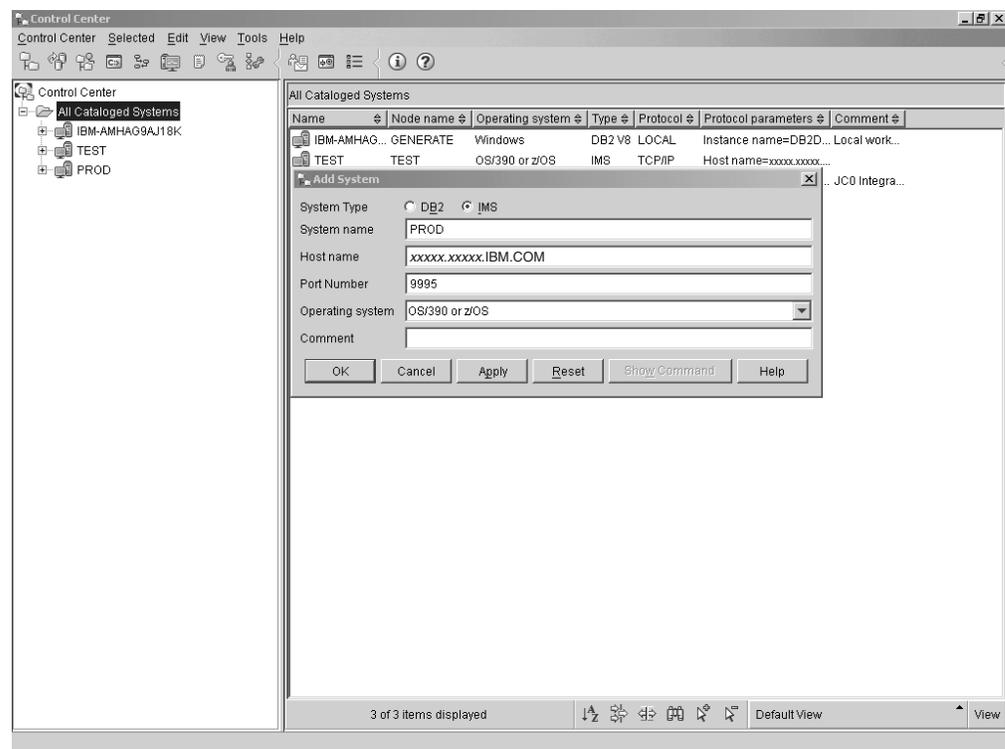


Figure 30. Example of the Control Center Add System dialog

3. Started the Command Center by clicking **Tools** → **Command Center** from the menu bar.
-
4. In the Command Center:
 - a. In the **Command type** field, selected **IMS commands** from the pull-down menu
 - b. In the **IMS sysplex** field, selected **PROD** (the name of our IMSplex) from the **Select Connection** dialog
 - c. When prompted, entered the user ID and password that we had set during the installation of DB2 Control Center.

IMS Common Service Layer and the Single Point of Control

Example: Figure 31 is an example of the initial setup of the Command Center: **Add System** dialog box:

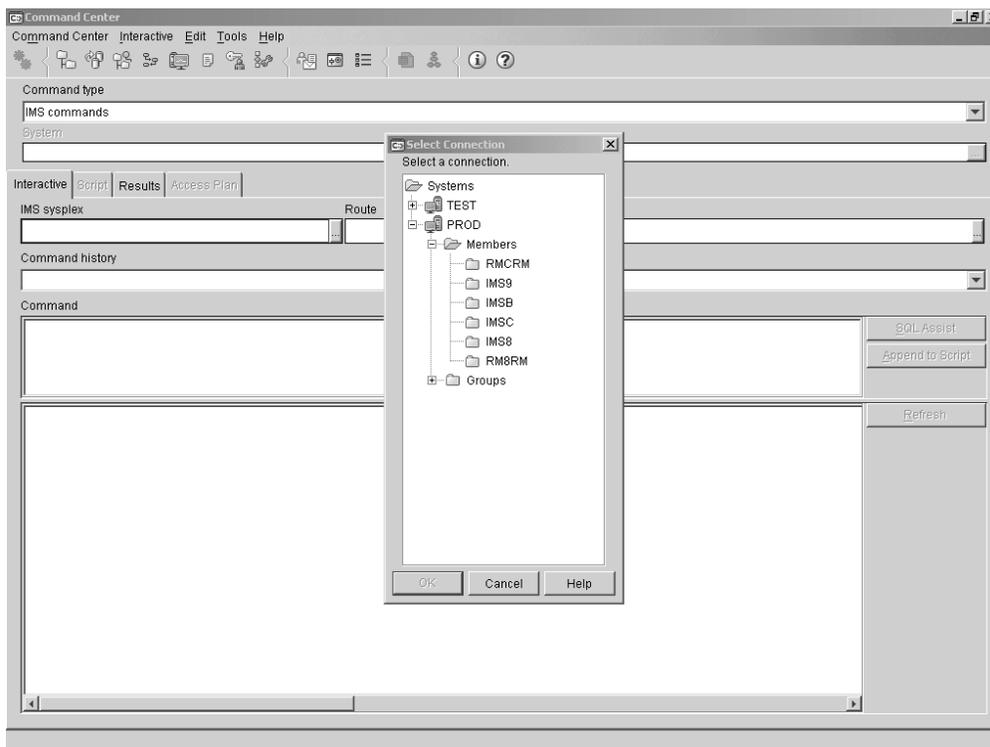


Figure 31. Example of the Command Center initial setup

5. To issue an IMS command:
 - a. In the **Route** field, selected the IMSplex member to which the command is to be issued
 - b. In the **Command** field, entered the IMS command to be issued
 - c. Clicked the **Execute** icon at the far left side of the icon bar, in the upper left corner

Example: Figure 32 on page 105 is an example of using the Command Center to issue the DIS QCNT LTERM MSGAGE 0 command to member IMSC in our PROD IMSplex:

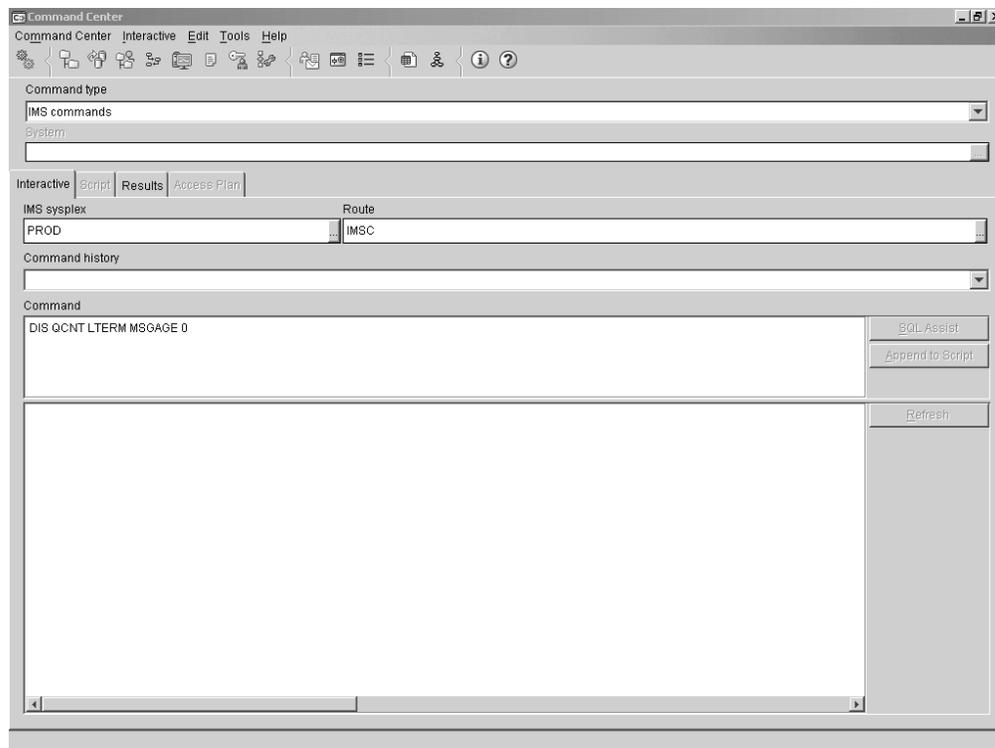


Figure 32. Example of issuing an IMS command to IMSplex member IMSC

Result: Figure 33 on page 106 is an example of the response to the DIS QCNT LTERM MSGAGE 0 command that was issued to member IMSC. Note that the response appears on a separate tab, **Results**, in the Control Center display.

IMS Common Service Layer and the Single Point of Control

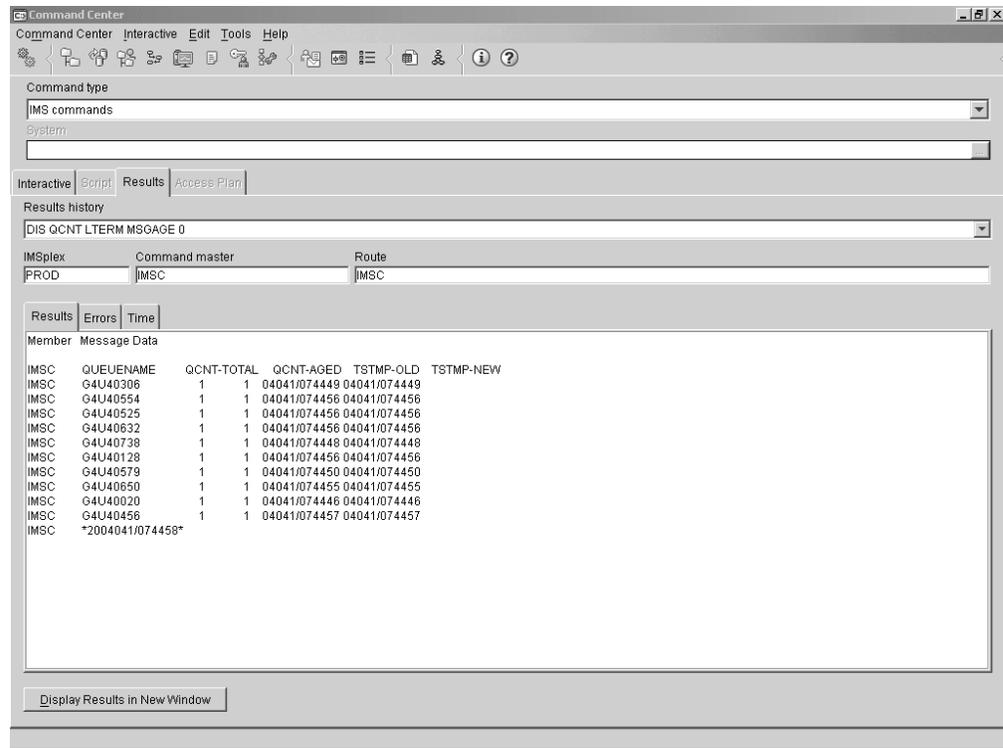


Figure 33. Example of the response to an IMS command that was issued to IMSplex member IMSC

Example: Figure 34 on page 107 is an example of issuing the DIS LINE 1 command to all members of the IMSplex by selecting All_Members in the Route field:

IMS Common Service Layer and the Single Point of Control

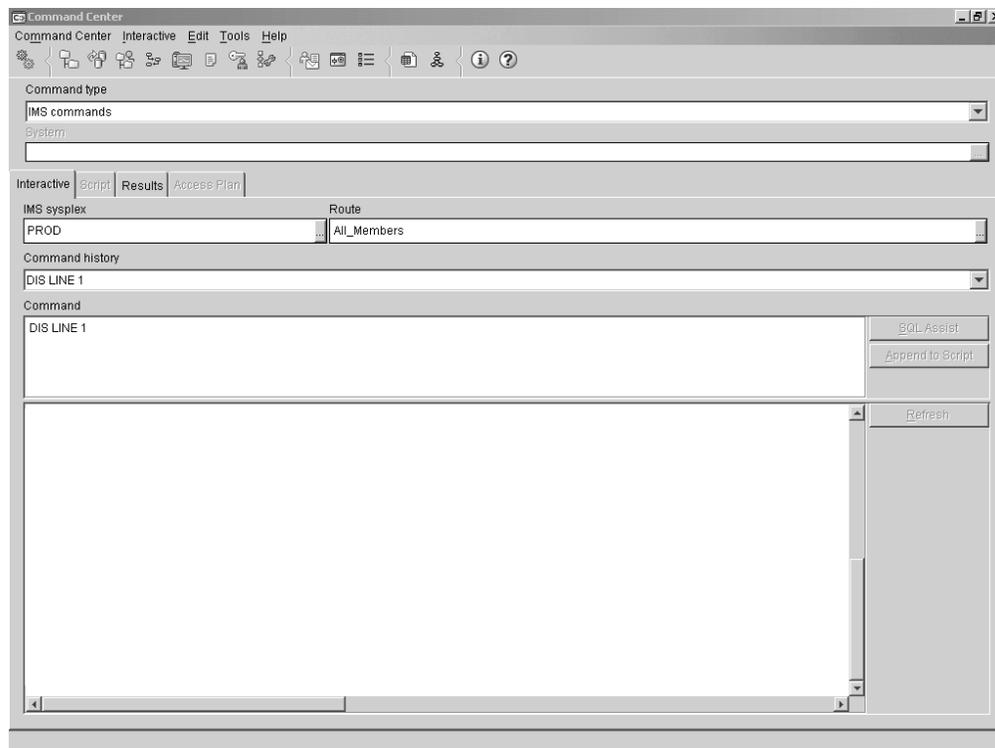


Figure 34. Example of issuing an IMS command to all members of the IMSplex

Result: Figure 35 on page 108 is an example of the response to a command that was issued to all members of the IMSplex:

IMS Common Service Layer and the Single Point of Control

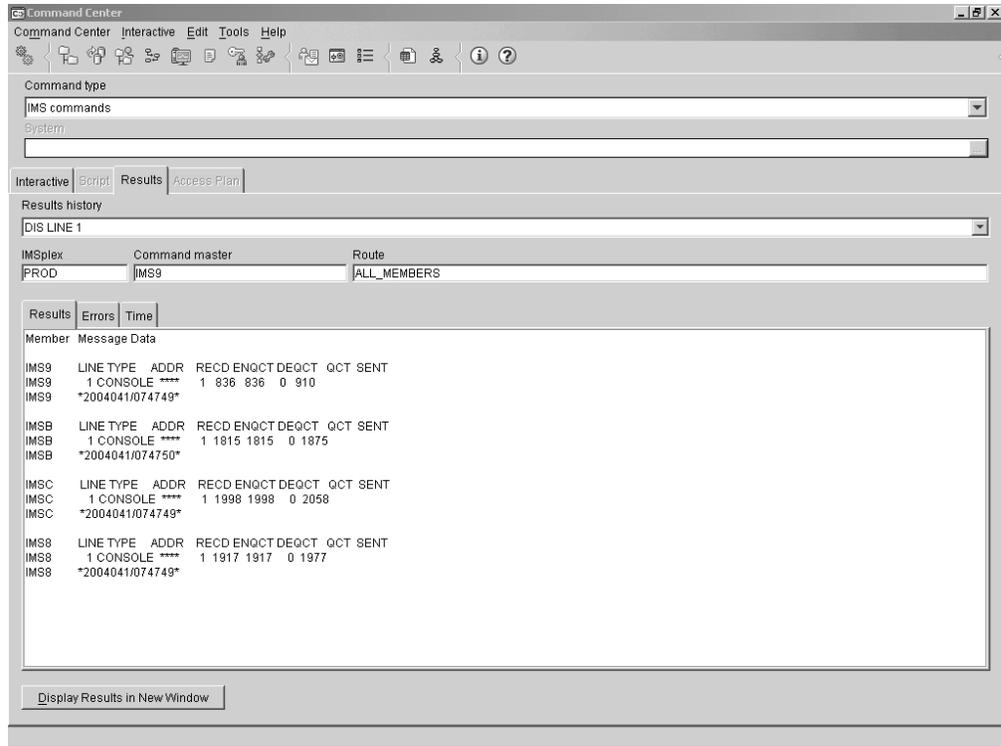


Figure 35. Example of the response to an IMS command that was issued to all members of the IMSplex

Chapter 8. Parallel Sysplex automation

In this chapter, we typically describe how we use automation to more efficiently operate our sysplex from a single point of control, and to automate the startup, shutdown, and restart of many of our major subsystems and applications.

Our early experiences with automation

We began writing about Parallel Sysplex automation in our 1997 test reports. At that time, we were just beginning to use NetView and System Automation for OS/390 (then called SA/MVS) to more efficiently operate our sysplex. We were running NetView V3R1 and SA/MVS V1R2.

We eventually migrated to Tivoli NetView for OS/390 V1R4 and System Automation for OS/390 (SA OS/390) V1R3, including the sysplex automation enhancements that IBM delivered in October, 1999, as an SPE in APAR OW39485. For information about our use of those products, see our December 2001 edition.

Automation with msys for Operations

Managed System Infrastructure for Operations (msys for Operations), which was introduced as a new base element in z/OS V1R2, simplifies the day-to-day operation of z/OS and z/OS.e Parallel Sysplex configurations by automating typical operator tasks and events. msys for Operations actually includes parts of two licensed standalone products: Tivoli NetView for OS/390 and System Automation for OS/390. We tested and ran msys for Operations along with Tivoli Netview for OS/390 V5R1 and SA OS/390 V1R3, prior to migrating to SA OS/390 V2R3.

Migrating to System Automation for OS/390 Version 2 Release 3

We have now completed our migration from msys for Operations to SA OS/390 V2R3. We followed the instructions in *System Automation for z/OS Planning and Installation*, SC33-7038, in the appendix about migrating to SA OS/390 from msys for Operations. Also, for basic information, we found *System Automation for z/OS User's Guide*, SC33-7040, to be very useful. You can find the SA OS/390 publications on the z/OS Internet Library at www.ibm.com/servers/eserver/zseries/zos/bkserv/ or under the **Library** link on the System Automation Web site at www.ibm.com/servers/eserver/zseries/software/sa/.

We continue to run Tivoli Netview for OS/390 V5R1. We also continue to run the NetView focal points, as we have discussed in previous years' test reports, on our current release of NetView. (See our December 2001 edition for more information about our NetView focal points.)

Applying service for SA OS/390: We needed to apply the fixes for the following SA OS/390 APARs: OA04046, OA04152, and OA05945.

Migrating the contents of our AOFCUST member: Our migration included using the INGCUST dialog to migrate the contents of our AOFCUST member to automation control file (ACF) fragments. As we reported on previously, we had customized our AOFCUST member to define a set of spare, local page data sets that can automatically be added when we experience an auxiliary storage shortage (see our December 2003 edition).

Parallel Sysplex automation

Starting the automation manager: In our sysplex, we use a startup procedure called HSAMPROC to start the automation manager. In our environment, we always start automations using a cold start. We generally do not use automations to control DB2, CICS, and IMS because the nature of our testing often requires that we vary the start and stop times for those subsystems, rather than leave them on a regular schedule.

We experienced no problems in migrating to System Automation for OS/390 Version 2 Release 3. We are continuing to run with it.

Using SA OS/390

Our operations staff invokes the INGPLEX command in full mode to access the main menu of sysplex-related functions in SA OS/390.

Using the DRAIN and ENABLE subcommands

From the INGPLEX main menu, we make frequent use of the DRAIN subcommand for clearing off our coupling facilities for maintenance. We find that this function saves us considerable time in our coupling facility maintenance activities.

Currently, when the DRAIN function completes, the coupling facility status changes to DRAINED NOHWACC because we do not have the BCP internal interface working. We then use the HMC to manually deactivate and re-activate the drained coupling facility.

After re-activating the coupling facility, we use the ENABLE subcommand to repopulate it with structures.

Refreshing the automation manager

Each time you make changes to the ACF, you must refresh the automation manager. Failure to refresh the automation manager following an ACF build results in the following message:

```
AOF618I NO VALID ACF FOUND FOR sysname - ACF TOKEN MISMATCH
```

This means that the ACF does not have the same token as the automation manager's configuration file.

We perform the following steps to refresh the automation manager after making changes to the current ACF:

1. From a NetView agent session, issue the following command:

```
INGAMS
```

Result: The following is an example of the dialog panel that appears:

```

Session C - [24 x 80]
File Edit View Communication Actions Window Help
-----
INGKYAMO SA 0S/390 - Command Dialogs Line 13 of 28
Domain ID = PETJ8 ----- INGAMS ----- Date = 02/12/04
Operator ID = BOBBYG Sysplex = UTCPLXJ8 Time = 07:22:18

Cmd: A Manage B Show Details C Refresh Configuration D Diagnostic

Cmd System Member Role Status Sysplex XCF-Group Release Comm
-----
- JH0 JH0 AGENT READY UTCPLXJ8 INGXSG V2R2M0 XCF
- JH0 JH0$$$$$1 SAM READY UTCPLXJ8 INGXSG V2R2M0 XCF
- J80 J80 AGENT READY UTCPLXJ8 INGXSG V2R2M0 XCF
- J80 J80$$$$$1 SAM READY UTCPLXJ8 INGXSG V2R2M0 XCF
- J90 J90 AGENT READY UTCPLXJ8 INGXSG V2R2M0 XCF
- J90 J90$$$$$1 SAM READY UTCPLXJ8 INGXSG V2R2M0 XCF
- TPN TPN AGENT READY UTCPLXJ8 INGXSG V2R2M0 XCF
- c TPN TPN$$$$$1 PAM READY UTCPLXJ8 INGXSG V2R2M0 XCF
- Z0 Z0 AGENT READY UTCPLXJ8 INGXSG V2R2M0 XCF
- Z0 Z0$$$$$1 SAM READY UTCPLXJ8 INGXSG V2R2M0 XCF
- Z1 Z1 AGENT READY UTCPLXJ8 INGXSG V2R2M0 XCF
- Z1 Z1$$$$$1 SAM READY UTCPLXJ8 INGXSG V2R2M0 XCF

Command ==>
PF1=Help PF2=End PF3=Return PF6=Roll
PF7=Back PF8=Forward PF9=Refresh PF12=Retrieve

MA c 17/003
    
```

Figure 36. Example of the INGAMS dialog

2. Locate the entry for the system that is acting as the primary automation manager (PAM). The entry will indicate PAM in the Role column. In the above example dialog, system TPN is acting as the PAM.
3. Enter the C (refresh configuration) line command next to the PAM system and press Enter.
Result: The following is an example of the dialog panel that appears:

Parallel Sysplex automation

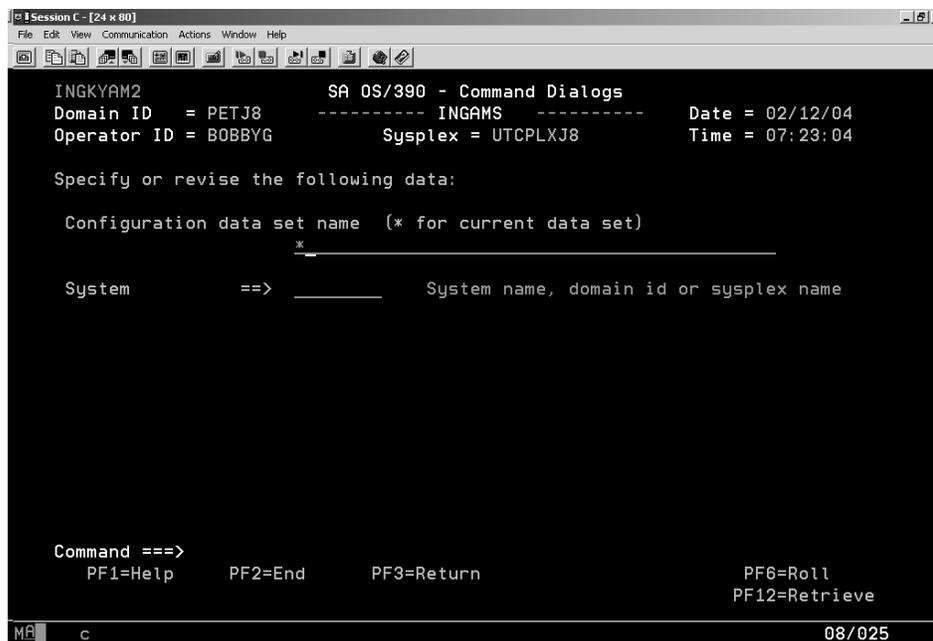


Figure 37. Example of the Refresh Configuration dialog

4. Enter an asterisk (*) for the configuration data set name and press Enter.

This refreshes the automation manager's configuration from the updated ACF.

Turning off the automation flag for a resource

During our testing, we often need to turn off automations for various resources so that we can manually control them. This turns the automation flag off in the automation manager, not the automation agent. The automation manager will remember the flag's setting unless all automation managers are brought down and restarted COLD.

We perform the following steps to turn off the automation flag for a resource:

1. From the NetView console, enter the following command:

```
DS subsystem
```

Example: To display the LDAP servers, we would issue the following command:

```
DS LDAP*
```

Result: The following is an example of the dialog panel that appears:



Figure 38. Example display from the DS LDAP* command

2. Enter the A (update) line command next to the desired resource and press Enter.
Example: To update the automation settings for the LDAPSRV server, enter an A next to that resource.
Result: The following is an example of the dialog panel that appears:

Parallel Sysplex automation

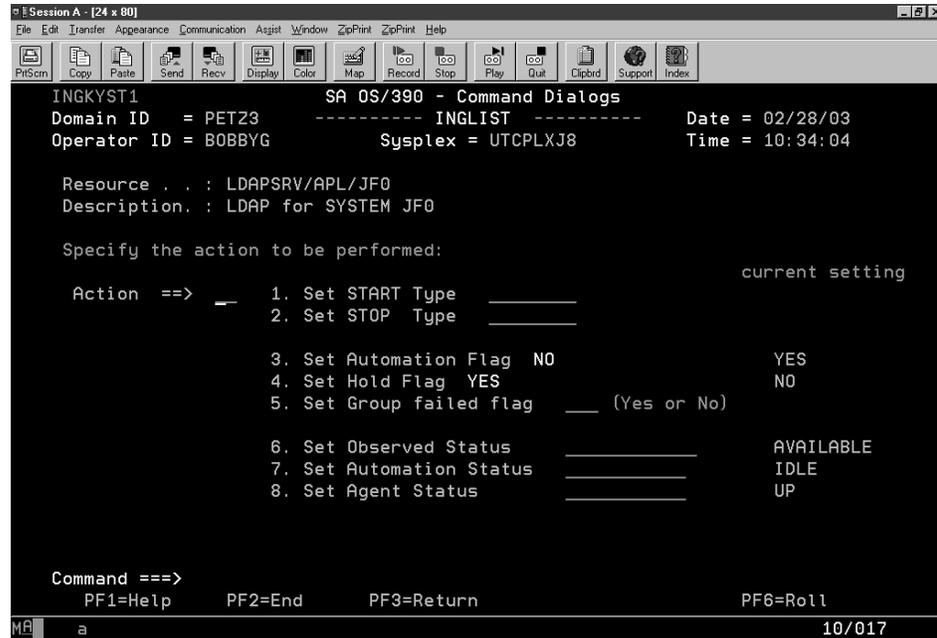


Figure 39. Example of the automation settings dialog for the LDAPSRV server (automation flag is on)

Note that the value under the **current setting** column for the automation flag is YES. This means that automation is turned on for this resource.

3. To turn off the automation flag, enter 3 (Set Automation Flag NO) on the **Action** line and press Enter.

Result: The following is an example of the dialog panel that appears:

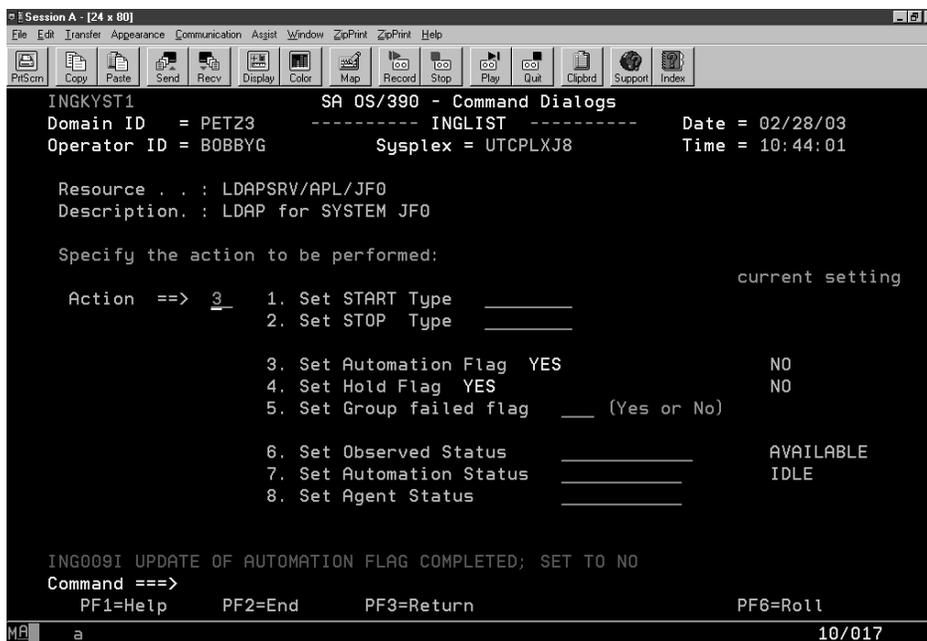


Figure 40. Example of the automation settings dialog for the LDAPSRV server (automation flag is off)

Note that the value under the **current setting** column for the automation flag is now NO. This means that automation is turned off for this resource.

The resource can now be manually started and stopped.

Chapter 9. Testing SPE Console Restructure (APAR OA09229)

We installed and tested with the V1R6 level of APAR OA09229 (concerning SPE Console Restructure). Please review the APAR description to see if you are experiencing the reported problem.

Part 2. Networking and application enablement

Chapter 10. About our networking and application enablement environment	123
Our networking and application enablement configuration	123
Our Ethernet LAN configuration	124
Our ATM configuration	125
Our IPv6 Environment Configuration	125
z/OS UNIX System Services changes and additions	125
TCPIP Profile changes	126
Dynamic XCF addition	126
Dynamic VIPA additions	126
OMPROUTE addition	126
NAMESERVER changes	127
Forward file changes	127
Reverse file entry addition	127
Our token ring LAN configuration	127
More about our backbone token ring	128
What's happening in LAN A?	128
What's happening in LAN B?	129
What's happening in LAN C?	130
Comparing the network file systems	132
Networking and application enablement workloads	132
Enabling NFS recovery for system outages	133
Setting up the NFS environment for ARM and DVIPA	133
Step for setting up our NFS environment	134
Chapter 11. Using z/OS UNIX System Services	137
z/OS UNIX enhancements in z/OS V1R5	137
Remounting a shared HFS	137
Mounting file systems using symbolic links	137
Creating directories during z/OS UNIX initialization	138
Testing the MKDIR keyword	139
Testing the SYNTAXCHECK keyword	140
Temporary file system (TFS) enhancements	141
Overview of the TFS enhancements that we tested	141
Testing the TFS enhancements	143
z/OS UNIX enhancements in z/OS V1R6	145
Using multipliers with BPXPRMxx parameters	146
Testing the multipliers	146
Using the superkill option	146
Using wildcard characters in the automove system list (SYSLIST)	148
Using the clear and uptime shell commands	149
Using the clear command	149
Using the uptime command	149
Enhanced latch contention detection	150
Testing contention recovery	150
Shells and utilities support for 64-bit virtual addressing	151
Overview of 64-bit support	151
Examples of the utilities that we tested	152
Using distributed BRLM	159
Using ISHELL enhancements	161
Using the hierarchical file system (HFS)	164
Automount enhancement for HFS to zSeries file system (zFS) migration	164
Using the zSeries file system (zFS)	165

	zFS enhancements in z/OS V1R6	165
	zFS parmlib search	165
	zFS performance monitoring with zfsadm (query and reset counters)	166
	HANGBREAK, zFS modify console command	168
	Issuing the su command and changing TSO identity.	169
	Removing additional diagnostic data collection from OMVS CTRACE LOCK	
	processing	169
	Chapter 12. Using the IBM HTTP Server	171
	Using gskkyman support for storing a PKCS #7 file with a chain of certificates	171
	Chapter 13. Using LDAP Server	173
	Overview of our LDAP configuration.	173
	Setting up the LDAP server for RACF change logging	174
	Activating change notification in RACF.	175
	Setting up the GDBM backend for the LDAP server	175
	Testing the change logging function and the GDBM database	177
	Searching the GDBM database	177
	Testing the maximum number of change log entries	177
	Searching the GDBM database anonymously	180
	Deleting change log entries	183
	Using the z/OS LDAP client with the Windows 2000 Active Directory service	183
	Using LDAP with Kerberos authentication	184
	Problems we experienced with our workload	184
	Abend 0C6 in LDAP Server.	184
	Abend 0C4 in gss_release_buffer in z/OS LDAP client	185
	Setting up SSL client and server authentication between z/OS LDAP V1R6	
	server/client and Sun ONE Directory Server 5.2 server/client.	186
	Setting up SSL client and server authentication between z/OS LDAP V1R6	
	server/client and IBM Tivoli Directory Server 5.2 server/client	192
	LDAP Server enhancements in z/OS V1R6	196
	LDAP migration to z/OS V1R6.	196
	Setting up a peer-to-peer replication network between an IBM Tivoli	
	Directory Server 5.2 and a z/OS LDAP Server	197
	Configuration Option 1	197
	Configuration Option 2	200
	Reference information	203
	Using DB2 restart/recovery function.	203
	Migrating to DB2 V8	204
	Module DSNAOCLI was not found in an authorized library	204
	Plans needed to be rebound using SQLERROR(CONTINUE)	204
	Using alias support	205
	Using the enhanced LDAP configuration utility (LDAPCNF).	206
	Using change logging with TDBM	207
	Chapter 14. Using Kerberos (Network Authentication Service)	211
	Setting up a Kerberos peer trust relationship between z/OS and Windows 2000	211
	Enabling the peer trust relationship on z/OS.	211
	Defining the Windows 2000 realm to the Kerberos server on z/OS	211
	Defining the cross-realm certification in RACF	212
	Testing the peer trust relationship	212
	Network Authentication Service (NAS) enhancements in z/OS V1R6.	213
	Accessing SYS1.SIEALNKE	213
	FTP with Kerberos	214
	Where to find more information	214
	FTP server enablement for Kerberos	214

	Assigning service principals	214
	The ftp.data file for the FTP Server	215
	Adding the keytab file	215
	Running without a keytab file	216
	Configuring a Linux workstation for Kerberos	216
	Creating the ftp.data file for the z/OS client ftp user	216
	Testing FTP with Kerberos	217
	Problems encountered	217
	Working with Kerberos principals in RACF	217

Chapter 15. Using the IBM WebSphere Business Integration family of products.

	Using WebSphere MQ shared queues and coupling facility structures	219
	Our queue sharing group configuration	219
	Our coupling facility structure configuration	219
	Testing the recovery behavior of the queue managers and coupling facility structures	220
	Queue manager behavior during testing	220
	Suggested MQ maintenance	221
	Additional experiences and observations	221
	Implementing WebSphere MQ shared channels in a distributed-queuing management environment	222
	Our shared channel configuration	223
	Shared inbound channels	223
	Shared outbound channels	224
	Testing shared channel recovery	224
	Testing channel initiator failure	225
	Testing queue manager failure	225
	Testing DB2 failure	225
	Using WebSphere Business Integration Message Broker	226
	Testing WMQI V2.1 on DB2 V8	226
	Setting the _BPXK_MDUMP environment variable to write broker core dumps to MVS data sets	226
	Resolving a EC6–FF01 abend in the broker	228
	Migrating WebSphere MQ Integrator V2.1 to WebSphere Business Integration Message Broker V5.0	228
	Migration activities on the Windows platform	228
	Migration activities on the z/OS platform	228
	Applying WBIMB V5.0 Fix Pack 02 and Fix Pack 03.	229
	Some useful WBIMB Web sites	229

Chapter 16. Using IBM WebSphere Application Server for z/OS 231

	About our z/OS V1R6 test environment running WebSphere Application Server	231
	Our z/OS V1R6 WebSphere test environment	231
	Current software products and release levels	231
	Our current WebSphere Application Server for z/OS configurations and workloads	232
	Other changes and updates to our WebSphere test environment	235
	Migrating WebSphere Application Server for z/OS JDBC from DB2 V7 to DB2 V8	235
	Using DB2 UDB JCC Connectors	235
	Migrating to CICS Transaction Gateway Connector V5.1	236
	Installing CTG 5.1	236
	Updates to CTG Daemon startup procs	236
	Installing J2EE CICS ECI Connector in WebSphere Application Server V 5.1	236

References	236
Enabling Global Security and SSL on WebSphere Application Server for z/OS	236
Global Security — "the Big Switch"	237
Migrating from WebSphere Application Server for z/OS 5.0.2 to 5.1 with Global Security enabled	237
References	238
Using the WebSphere Application Server for z/OS 5.x plug-in for HTTP Server and Sysplex Distributor with our WebSphere Application Server for z/OS J2EE Servers	238
Using the HTTP Server with WebSphere Application Server for z/OS plug-in along with our J2EE Servers	239
Bottlenecks in our HTTP Server	240
Scaling up the HTTP Server with WebSphere Application Server for z/OS plug-in along with our J2EE Servers	241
TrustedProxy setting in J2EE Server's WebContainer	242
HTTP Server SSL setup needs J2EE Server's CA	242
Customizing the WebSphere Application Server for z/OS plug-in configuration file (pluginconf.xml)	243
Improving Static Content performance	244
Sysplex Distributor usage with HTTP Server / WebSphere Application Server for z/OS J2EE Servers	246
Where to find more information	248
Specific documentation we used	248
Chapter 17. Using EIM authentication	251
Client authentication using digital certificates	251
Resolving problems during our testing	251
Testing the client authentication using digital certificates	252
Kerberos authentication	252
Clearing up a documentation inaccuracy	253
Testing the Kerberos authentication	253
CRAM-MD5 password protection	254
EIM enhancements in z/OS V1R6	254
x.509 certificate registries	254
Testing associations	255
Testing Filtering	256
Create an x.509 certificate filter policy using a certificate	257

The following chapters describe the networking and application enablement aspects of our computing environment.

Chapter 10. About our networking and application enablement environment

In this chapter we describe our networking and application enablement environment, including a high-level view of our configurations and workloads. We discuss networking and application enablement together because the two are greatly intertwined. You need the networking infrastructure in place before you can run many of the application enablement elements and features.

Our networking and application enablement configuration

Figure 41 illustrates at a high level our networking and application enablement configuration. In the figure, the broad arrows indicate general network connectivity of a given type, rather than specific, point-to-point, physical connections.

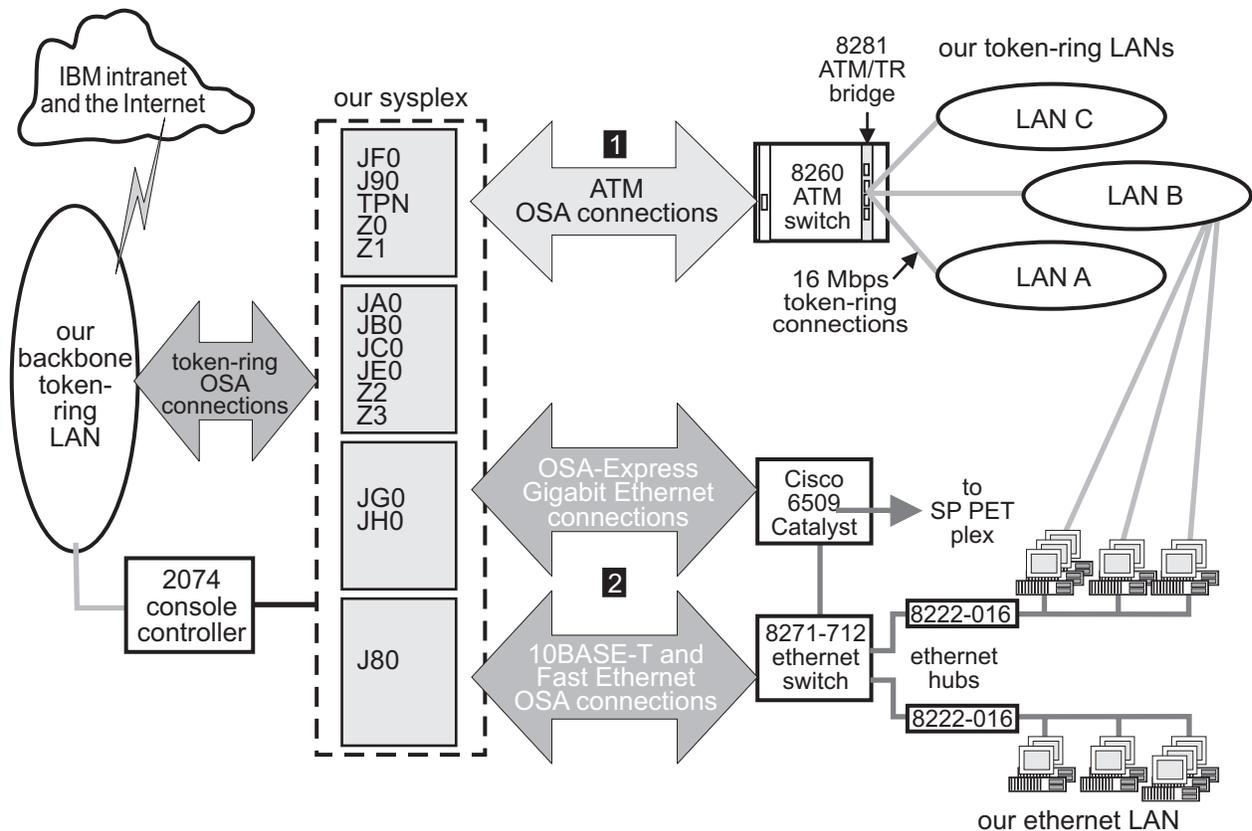


Figure 41. Our networking and application enablement configuration

Note the following about Figure 41:

- We use the following OSA features to connect our systems to our LANs:
 - OSA-2
ENTR (Ethernet/Token Ring)
 - OSA-Express
ATM (Asynchronous Transfer Mode)
FENET (Fast Ethernet)
Gigabit Ethernet (GbE)

Networking and applications environment

- All ATM connections (**1**) use ATM LAN emulation; we have no native ATM connections. Although not shown, some of our CPCs that do not have an ATM connection instead use OSA-2 ENTR to directly connect to each of our token-ring LANs.
- Host system Ethernet connections (**2**) use either OSA-2 ENTR 10BASE-T, OSA-Express FENET, or OSA-Express Gigabit Ethernet features depending on the CPC model and the type of adapter it supports.
- Although not shown, all RS/6000s on LAN A also have a direct connection to the backbone token ring.
- We recently replaced our Gigabit Ethernet switch with a Cisco 6509 Catalyst switch. This new switch handles all of our Gigabit Ethernet and some of our Ethernet connections. Eventually, all of our Ethernet traffic will flow through the Cisco 6509 and we'll remove the 8271-712 Ethernet switch. (For technical details about the Cisco 6500 Catalyst family, go to <http://www.cisco.com>.)
- We also added Gigabit Ethernet connectivity between our sysplex and a remote AIX cluster owned by the SP PET team. We use this connectivity for the AIX portion of our bookstore application.

For an illustration of our VTAM configuration, see “Our VTAM configuration” on page 19.

If you are familiar with our test reports, then you know that we have always described our networking configuration in exacting detail, as we felt that the complexity of our environment required a great deal of explanation. For example, at one time we were very specific about which of our system images could access which of our network resources. However, as we progress, we are concentrating more and more on TCP/IP and expanding our use of Ethernet. As a result, things are becoming more similar than dissimilar and connectivity between our host systems and network resources is approaching any-to-any.

Accordingly, we have shifted our networking discussion to a somewhat more conceptual level and focus on how our infrastructure enables us to test and exploit new features and functions. We will continue to highlight specific aspects of our configuration as significant changes occur and we introduce new technologies.

Our Ethernet LAN configuration

Our network configuration includes an Ethernet LAN. We primarily use it for FTP testing from Windows 95 and Windows NT clients and for VIPA testing. (For more information about VIPA, see our December '99 edition.) Many of our Ethernet client workstations also contain a token-ring adapter that connects the workstations to our token-ring LAN B as well. We use an OS/2 LAN Server on LAN B to drive the FTP testing on the Ethernet clients. You can read more about this setup in “What’s happening in LAN B?” on page 129

Our systems’ Ethernet connectivity includes a combination of 10BASE-T, Fast Ethernet, and Gigabit Ethernet connections using OSA-2 ENTR, OSA-Express FENET, and OSA-Express Gigabit Ethernet features, respectively.

Note that the connections between our OSA-Express Gigabit Ethernet features and our Cisco 6509 Catalyst switch operate at 1000 Mbps. The Fast Ethernet connections between our OSA-Express FENET features and our 8271 Ethernet switch, as well as those between our Cisco 6509 and the 8271, operate at 100 Mbps. The 10BASE-T connections from the 8271 to the 8222 Ethernet hubs and client workstations operate at 10 Mbps.

The OSA-Express FENET feature operates at either 10 or 100 Mbps in half- or full-duplex mode and supports auto-negotiation with its attached Ethernet hub, router, or switch. We used the latest edition of *zSeries OSA-Express Customer's Guide and Reference* and the OSA/SF GUI for Windows to install and configure the OSA-Express FENET feature. (See our December 1999 edition for our experiences installing the OSA/SF GUI for Windows.) We also recommend that you check with your IBM support representative to ensure that you have the latest microcode level for this feature.

Our ATM configuration

As we note above, our configuration includes OSA-Express ATM features operating in LAN emulation mode only. Therefore, when you see the term *ATM* in this chapter, understand it to mean *ATM LAN emulation*. (See our December 1998 edition for details on our ATM implementation.)

We use ATM for high-speed, bi-directional, asynchronous connectivity between our z/OS systems and our 8260 ATM switch. The 8260 then connects to the 8281 LAN bridge and provides access to all three of our token-ring LANs. The ATM links operate at 155 Mbps while the token-ring LANs still operate at 16 Mbps. Therefore, the maximum combined token-ring traffic from all three LANs is only 64 Mbps, which *each* ATM link easily accommodates.

Note that because of the wide variety of hardware we employ, not every CPC in our sysplex has an ATM connection. For those CPCs that do not, we use OSA-2 ENTR to provide direct connections to each of our token-ring LANs. Either way, it's all transparent to the end user.

Our ipV6 Environment Configuration

With z/OS V1R6, we now have an ipV6 environment equivalent to our ipV4 environment. V1R6 now supports OSPF V3 for ipV6 and ipV6 support for DVIPA and Sysplex Distributor.

We used the following manuals as guides in setting up ipV6.

- *z/OS Communications Server: IP Configuration Guide*, SC31-8775
- *z/OS Communications Server: IP Configuration Reference*, SC31-8776
- *z/OS Communications Server: IPv6 Network and Application Design Guide*, SC31-8885

To configure a z/OS image for ipV6 the following changes have to be made:

Note: This is not meant to be an all inclusive guide for ipV6 setup.

1. Add new NETWORK, AF_INET6 to BPXPRMxx statement.
2. Add New INTERFACE to TCPIP profile for ipV6 'device'.
3. Add support for DYNAMIC XCF
4. Create DVIPA for ipV6
5. Add INTERFACE to OMPROUTE profile.
6. Make appropriate additions to Nameserver.

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

Networking and applications environment

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 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 stack that supports IPv4 and IPv6

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

```
FILESYSTYPE TYPE(CINET) ENTRYPPOINT(BPXCINT)
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) ENTRYPPOINT(EZBPFINI)
SUBFILESYSTYPE NAME(TCPCS2) TYPE(CINET) ENTRYPPOINT(EZBPFINI)
SUBFILESYSTYPE NAME(TCPCS3) TYPE(CINET) ENTRYPPOINT(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:

Note: V6Z2FTP is the INTERFACE name for this VIPA.

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

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.

NAMESEVER 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  ZOEIP.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  ZOEIP.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.
```

Our token ring LAN configuration

As Figure 41 illustrates, we have a total of four token-ring LANs: a backbone ring and three test LANs that use various:

- Communications protocols (TCP/IP, SNA, NetBIOS, and Internet Packet Exchange (IPX))
- Workstation operating systems (AIX, Linux, OS/2, PC DOS, and Microsoft Windows NT, Windows 95, and Windows 2000)
- Workstation types (RS/6000s and various types of PCs)

LANs A, B, and C in Figure 41 use only the token-ring LAN protocol. The three LANs connect to our host systems through the 8281 LAN bridge and 8260 ATM switch as described above. (You can read about our ATM experiences in our December 1998 edition.) For host systems running on CPCs that do not have an ATM connection, we instead use OSA-2 ENTR features to provide direct token-ring connections to each of the three LANs (these connections are not shown in Figure 41).

Note that we also have an OS/2 LAN Server with a CLAW protocol channel adapter that connects to system JE0 for LAN Server. This is not shown in Figure 41; see Figure 44 on page 131 for an illustration of this.

All of the systems in our sysplex can connect to the IBM SNA network using VTAM as long as either system Z0 or system J80 (the network node server) is available. In addition, all systems can get to the IBM TCP/IP network directly through our backbone token ring.

Networking and applications environment

We discuss how we use the backbone and LANs A, B, and C in greater detail in the following sections.

More about our backbone token ring

The token-ring backbone connects our test environment to the IBM corporate network or intranet and, beyond that, to the Internet. Rather than exist as an isolated entity, our ability to connect to the rest of the corporation and to the outside world yields us a much more viable and robust test environment. Some specific advantages include:

- We are able to access our network resources from our offices or while working from home, instead of having to be on the test floor all the time. This convenience and flexibility allows us to be more productive.
- We can perform more complete and realistic test scenarios with products and features, such as:
 - Tivoli Storage Management (TSM, formerly ADSM)
 - Firewall
 - NFS
 - Infoprint® Server
 - Rlogin
 - Telnet
 - Web access
- When we encounter a complex problem, we are able to have product developers from our local site and from other IBM locations work with us in our own test environment to help diagnose and resolve the problem.
- We keep our own documentation, such as test plans and run procedures, on our Web server and can access it from anywhere. As a result, we also implicitly test our networking environment just by performing our day-to-day administrative work.
- We install our configuration tools (for Firewall and OSA/SF, for example) on workstations attached to the backbone so that we can provide central access to the tools and share them across multiple systems.

For many of the same reasons, we also recently switched from running our RS/6000 workloads on LAN A to running them on the backbone, mostly to allow greater access to other resources and provide more realistic testing. See the next section for more about LAN A.

What's happening in LAN A?

Our RS/6000 workstations reside on LAN A but they also directly connect to the backbone. This additional connectivity allowed us to shift a majority of the workloads that we once performed exclusively on LAN A over to the backbone. LAN A itself still exists in our environment, but we don't use it for anything special from a functional standpoint.

Figure 42 on page 129 depicts our z/OS UNIX DCE test configuration in LAN A, including the connections from the RS/6000s to the backbone (which, for clarity, are not shown in Figure 41 above).

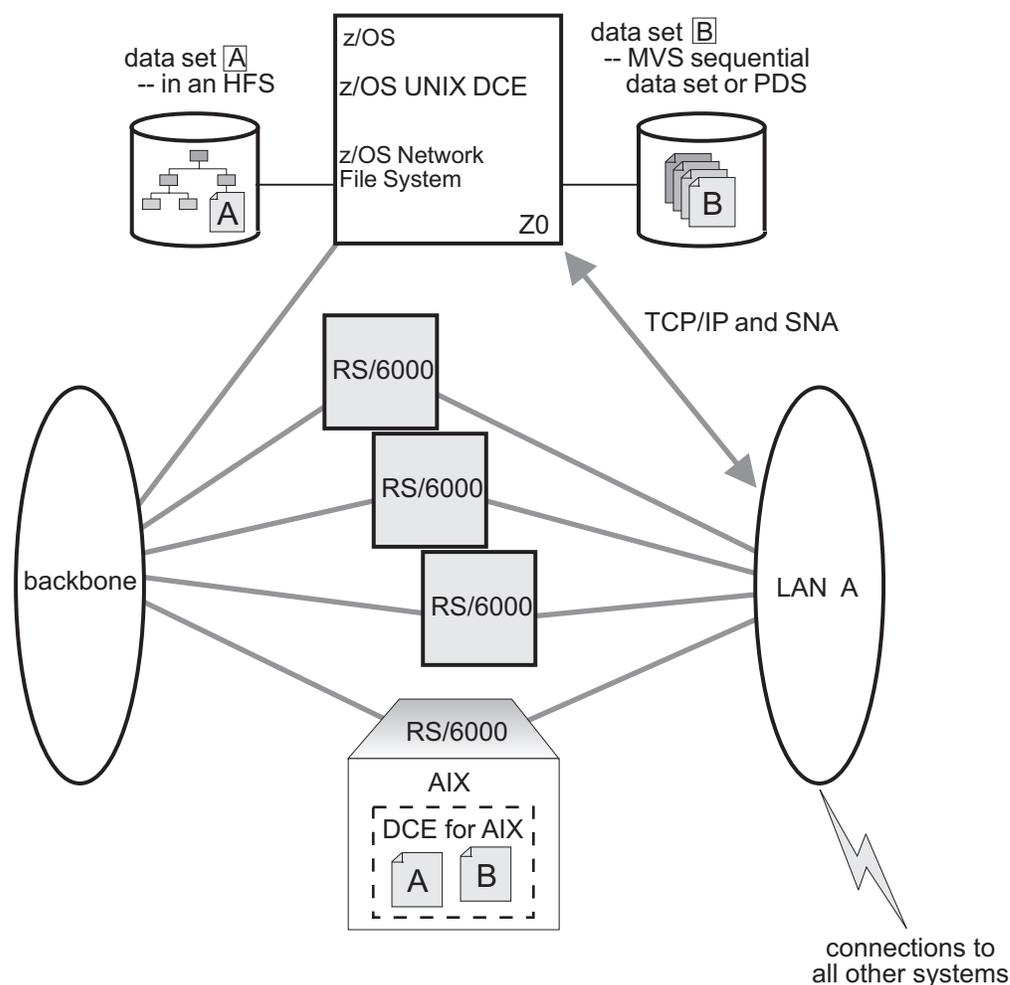


Figure 42. Our token-ring LAN A

The RS/6000 workstations on LAN A all run the AIX operating system. We use them to exercise the z/OS UNIX, DCE, and DFS™ functions. See “Our workloads” on page 20 for a more detailed description of these workloads. For more information about DCE and DFS, see “DCE and DFS Publications” in Appendix E, “Useful Web sites,” on page 321.

What’s happening in LAN B?

You might recall from our December 1996 edition that our LANs B and C started out as two functionally separate LANs. Later on, we combined their functionality and collectively referred to them as logical LAN BC. Well, we’ve now come full circle. For better performance and throughput, we are back to using LANs B and C as two functionally separate LANs.

LAN B has an OS/2 NFS function and an FTP function using TCP/IP. The OS/2 LAN Server on LAN B acts as a control workstation for our NFS and FTP workloads. The control data consists of the commands that start, stop, and otherwise regulate the execution of the workloads. The test data or workload data is the actual data that the workloads manipulate.

Networking and applications environment

The workstations that run the FTP workloads connect to both our token-ring LAN B and to our Ethernet LAN. The FTP control data comes from the OS/2 server to the clients over LAN B. The test data that the workloads manipulate travels over the Ethernet LAN.

The NFS function communicates with z/OS NFS using TCP/IP, and both the control data and the workload data travel over LAN B. (See “Comparing the network file systems” on page 132 for a description of the different types of NFSs we use.)

Figure 43 depicts our NFS and FTP test configuration in LAN B. For more information about NFS, see “Network File System Publications” in Appendix E, “Useful Web sites,” on page 321.

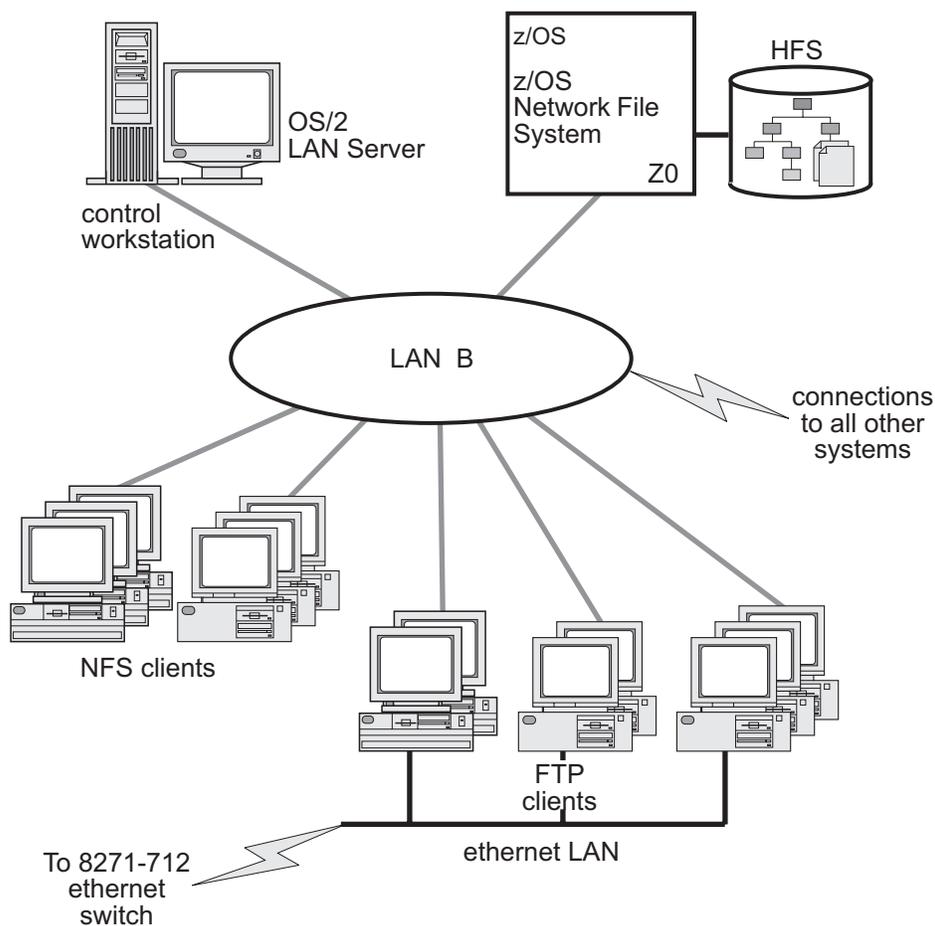


Figure 43. Our token-ring LAN B

What's happening in LAN C?

LAN C runs two different types of LAN Server scenarios using OS/2 LAN Servers as front-end processors (FEPs) to z/OS LAN Server. z/OS LAN Server expands the file storage capability of the OS/2 LAN Servers by storing workstation-format files in VSAM linear data sets on the z/OS host. These data sets are not readable by MVS users, but appear to the clients as though they are stored on the OS/2 LAN Servers.

First, we have an OS/2 LAN Server acting as a FEP (**A**) with a SNA connection to LAN Server in system Z0. We could conceivably connect the OS/2 LAN Server to

any z/OS system, but we currently happen to be using Z0. We use Communications Manager/2 for the SNA connection and APPC communications.

We also have another OS/2 LAN Server acting as a FEP (**B**) with a CLAW protocol connection to LAN Server in system JE0.

Typically, a LAN file server contains one or more large-capacity hard disk drives on which it stores files for access by the clients (or requesters). However, in our setup, the OS/2 LAN Servers do not store any workload-related programs or data on their own hard disks for use by the clients. All the workload-related programs and data reside on the z/OS system. This is completely transparent to the requesters, as they are only aware of the OS/2 LAN Servers which, in turn, interact with z/OS LAN Server on the host. The OS/2 servers do keep setup files, automation programs, and workstation configuration files on their own local disk drives.

Figure 44 depicts our LAN Server test configuration in LAN C. For more information about LAN Server, see “LAN Server Publications” in Appendix E, “Useful Web sites,” on page 321.

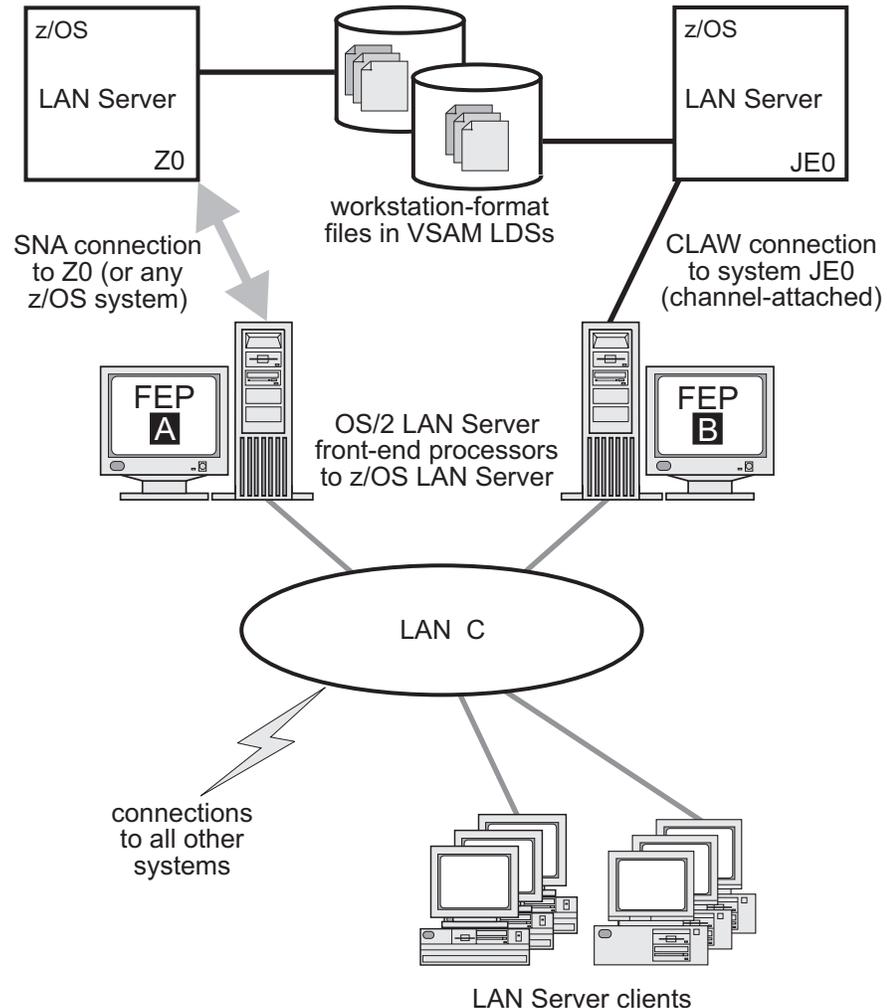


Figure 44. Our token-ring LAN C

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 allow 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 Customization and Operation*, SC26-7417 and *z/OS Network File System User's Guide*. In addition, you can read about WebNFS support in our December 1999 edition, and the use of the LAN Server NFS in our June 2004 edition. We hope to have additional experiences with these new functions to share with you in a future test report.

In the meantime, we'd like to highlight one aspect of the migration to the z/OS NFS. **Pay attention to the following words** in the section on allocating the mount handle data sets in *z/OS Network File System Customization and Operation*, SC26-7417: "Delete and allocate the mount handle data sets before running any new versions of the Network File System. If an old mount handle data set is used, the server issues a message and shuts down." We somehow missed this and attempted to migrate without deleting our old data sets and recreating them. When the server shut down, we had a difficult time figuring out why.

Note that APAR OW40134 recommends a change to the SHAREOPTIONS specified in the sample JCL for the IDCAMS job used to allocate the mount handle data sets. This sample JCL is both shipped in *hlq.NFSSAMP(GFSAMHDJ)* and illustrated in *z/OS Network File System Customization and Operation*. The sample JCL currently uses SHAREOPTIONS(3 3). However, the APAR instead recommends SHAREOPTIONS(1 3). While the sample code does work as it stands, it allows programs other than NFS to update the files. Using SHAREOPTIONS(1 3) limits the possibility of corruption to the mount handle database.

Networking and application enablement workloads

For information about our networking and application enablement workloads, see "Our workloads" on page 20.

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 45 on page 134: 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 45 on page 134 : 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.

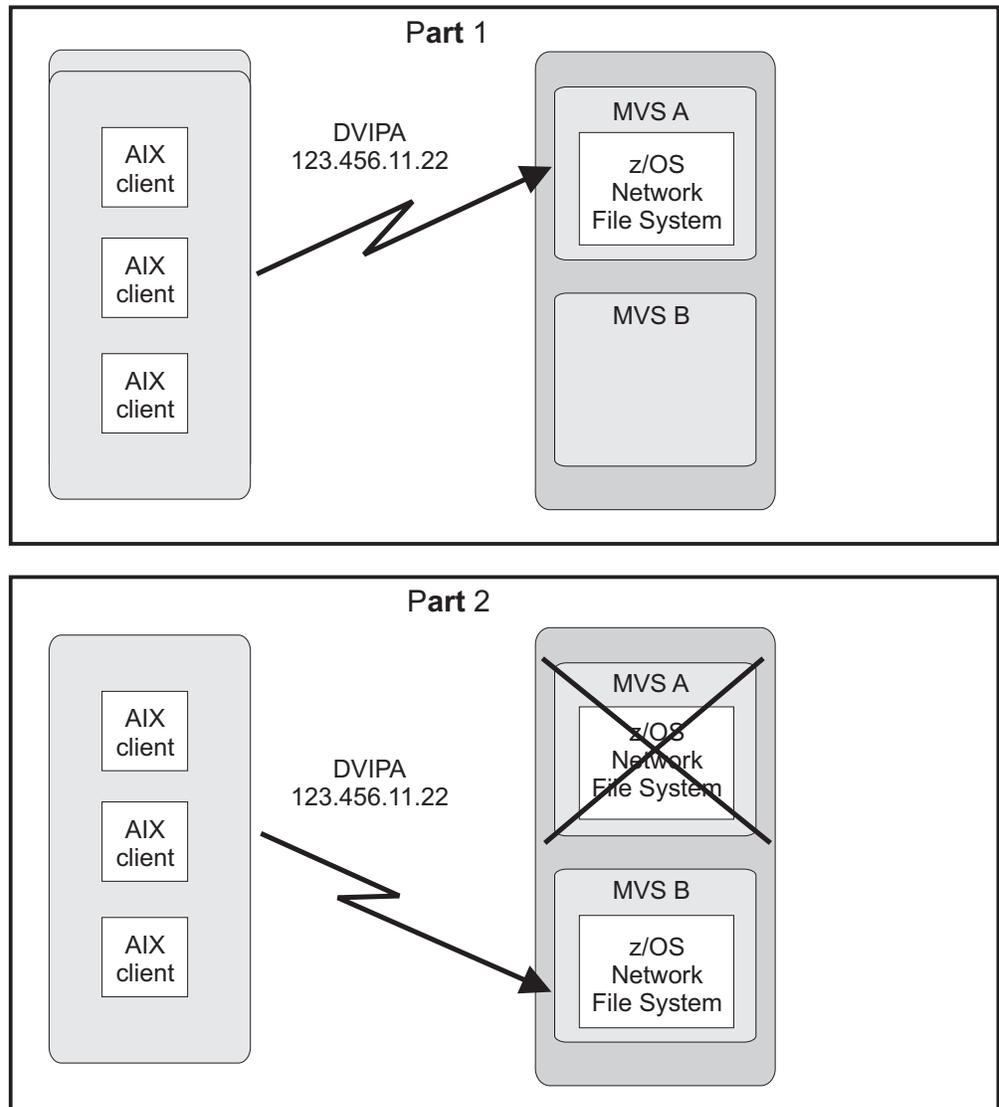


Figure 45. NFS configuration

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 MVS B 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 MVS B 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 IXARM.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')

```

Networking and applications environment

Chapter 11. Using z/OS UNIX System Services

In this chapter, we cover the following z/OS UNIX System Services topics:

- “z/OS UNIX enhancements in z/OS V1R5”
- “z/OS UNIX enhancements in z/OS V1R6” on page 145
- “Using the hierarchical file system (HFS)” on page 164
- “Automount enhancement for HFS to zSeries file system (zFS) migration” on page 164
- “Using the zSeries file system (zFS)” on page 165

z/OS UNIX enhancements in z/OS V1R5

z/OS UNIX made several enhancements in z/OS V1R5. In this section, we cover the following topics:

- “Remounting a shared HFS”
- “Mounting file systems using symbolic links”
- “Creating directories during z/OS UNIX initialization” on page 138
- “Temporary file system (TFS) enhancements” on page 141

We used the information in *z/OS UNIX System Services Planning* to help us plan and implement the above enhancements.

Remounting a shared HFS

Remounting a mounted shared HFS is a new function in z/OS V1R5 that allows you to remount an HFS or zFS file system within a directory tree so as to change the access mode. This function is also available for z/OS V1R4 by applying the fix for APAR OA02584.

We have utilized this new function to remount the version HFS from READ mode to RDWR mode. Our normal configuration is to have the version HFS mounted read-only, as IBM recommends. In our environment, we have a requirement to be able to make changes to /usr/lpp and other directories in the version HFS which, until z/OS V1R5, we could only accomplish during our STAGE3 processing (see the discussion of our CUSTHFS procedure in our December 2001 edition). While we still prepare the HFS for our configuration needs prior to implementing it into production, this new function provides us with the ability to perform additional changes while the HFS is being used in the production environment.

You can perform the remount from the ISHELL panel or by using the TSO unmount command from the server or any of the client systems within the same shared HFS sysplex. The following is an example of the TSO command for remounting our version HFS in read/write mode:

```
unmount filesystem('OMVSSPN.PETPA1.ROOT.FS') remount(rdwr)
```

Mounting file systems using symbolic links

z/OS V1R5 introduces support for using MVS system symbols in symbolic links. This provides the ability to mount different file systems at a logical mount point that resolves to a different path name on different systems. This function uses two new, special identifiers in a symlink, followed by an MVS standard symbolic string template. The special identifier indicates that the text that follows it requires symbolic substitution. We tested this function using HFS and zFS file systems.

The following are the new identifiers:

- \$SYSSYMR/*template*

Results in a relative path name. The path name lookup proceeds from its current position in the path name.

- `$SYSSYMA/template`

Results in an absolute path name. The path name lookup starts over from the root.

Note: When coding the `In` command, the new identifier ends with a forward slash (/) and the system symbol starts with a backslash (\).

To test the relative and absolute path name lookups, we used one of our existing system symbols, `&SYSCZONE`. The value of `&SYSCZONE` is "Z0" on our system Z0. The following examples illustrate the difference between the relative and absolute path name lookups.

Example: We issued the following `In` command in the OMVS shell on system Z0:

```
In -s \&SYSSYMR/\&SYSCZONE./testdir /pet5/rdir
```

On system Z0, we mounted `OMVSSPN.Z0.SYMBOLIC.TEST` at `/pet5/rdir`. The HFS was linked with a relative path name and was mounted at `/pet5/Z0/testdir`.

Example: We issued the following `In` command in the OMVS shell on system Z0:

```
In -s \&SYSSYMA/\&SYSCZONE./testdir /pet5/dir
```

On system Z0, we mounted `OMVSSPN.Z0.SYMBOLIC.TEST` at `/pet5/dir`. The HFS was linked with an absolute path name and was mounted at `/Z0/testdir`.

Creating directories during z/OS UNIX initialization

The z/OS UNIX parmlib member, `BPXPRMxx`, now supports a new, optional keyword, `MKDIR`, on the existing `ROOT` and `MOUNT` statements. The `MKDIR` keyword allows one or more directories to be created in the mounted file system as part of the mount process during z/OS UNIX initialization. You can specify multiple `MKDIR` keywords on each `ROOT` or `MOUNT` statement; the directories are created in the order in which the `MKDIR` keywords occur. Such directories can serve as mount points that can be used in subsequent `MOUNT` statements.

The `MKDIR` keyword has the following syntax:

```
MKDIR('pathname')
```

where *pathname* specifies a relative path name of a directory to be dynamically created after the file system has been successfully mounted. The path name must not start with a slash (/) and must be enclosed in single quotes.

The path name is relative to the file system's mount point (specified by the `MOUNTPOINT` keyword) and can contain intermediate directory components but each component must already exist in the file system hierarchy. You can use multiple `MKDIR` keywords to create the necessary intermediate directories. Note that the length of the `MKDIR` path name plus the length of the `MOUNTPOINT` path name must not exceed the value of the `PATH_MAX` configuration variable.

The directory to be created must reside in a file system that is mounted in `RDWR` mode. The directory will have permission bits of 755 and will inherit the `UID` and `GID` from its parent directory. These attributes will be overlaid when this directory is actually used as a mount point.

Note the following about the usage of the MKDIR keyword:

- Failure to create a directory does not cause the mount to fail. A message is written to the system log if there is a problem creating a directory. No message is written if the directory already exists.
- MKDIR is only supported in the BPXPRMxx parmlib member. There is no downlevel support for MKDIR; therefore, only use MKDIR in a common BPXPRMxx member when all sharing systems are at z/OS V1R5 or higher.
- File system reinitialization using the MODIFY BPXOINIT,FILESYS=REINIT command does not support the use of MKDIR in the BPXPRMxx member.
- The OMVS restart function (that is, MODIFY OMVS,SHUTDOWN followed by MODIFY OMVS,RESTART) does support the use of MKDIR in the BPXPRMxx member.
- MKDIR should not be used for file systems that mount asynchronously, such as the network file system (NFS). In such cases, the creation of the directory cannot be guaranteed. Message BPXF025I is issued to the system log when a file system is to be mounted asynchronously.
- Do not use MKDIR with the SYSNAME keyword when SYSNAME identifies a remote system to perform the mount, as the results are unpredictable. MKDIR will not process on a file system that is already mounted on a remote system.

Note also that, in addition to checking statement syntax, the SETOMVS SYNTAXCHECK=(xx) command also checks the MVS catalog for the existence of the HFS or zFS data set names used in each ROOT and MOUNT statement in the specified BPXPRMxx member. Messages are written to the syslog if any errors are found. Although mount points are not verified, this can help to ensure that mounts will succeed.

Testing the MKDIR keyword

We made the following changes to the mounts for the sysplex root (/) file system, each system-specific /tmp file system, and a zFS /pet3 file system in our common parmlib member, SYS1.PARMLIB(BPXPRM00):

```

ROOT FILESYSTEM('OMVSSPN.SYSPLEX.ROOT.FS') TYPE(HFS)
      MODE(RDWR) MKDIR('mkdirrootv1r5') A

MOUNT FILESYSTEM('OMVSSPN.&SYSNAME..TMP.FS') TYPE(HFS)
      MODE(RDWR) MOUNTPPOINT('/&SYSNAME./tmp') UNMOUNT
      PARM('FSFULL(90,5)') MKDIR('mkdirtmpv1r5') B

MOUNT FILESYSTEM('OMVSSPN.PET3.ZFS.FS') TYPE(ZFS) MODE(RDWR)
      MOUNTPPOINT('/pet3') AUTOMOVE(I,Z0,Z1,Z2,Z3) MKDIR('mkdirpet3v1r5') C

```

We issued the SETOMVS SYNTAXCHECK=(00) command to perform syntax and data set name checking on the member. The following message appeared:

```
IEE252I MEMBER BPXPRM00 FOUND IN SYS1.PARMLIB
```

We then checked the syslog to verify that there were no error messages.

We observed the following results the next time the systems were initialized:

A — **For the sysplex root (/) file system:** We had to wait until we could unmount the sysplex root (/) file system before the directory on the MKDIR keyword could be created the next time the root was mounted during z/OS UNIX initialization. The MKDIR on the ROOT statement will not process as long as the root file system is mounted on any system in the sysplex. Therefore, we tested this by taking down all of the systems in the sysplex and re-IPLing. The first system to join the sysplex mounted the root file system and successfully

processed the MKDIR keyword to create the /mkdirrootv1r5 directory. The directory had permission bits of 755 and had the same UID and GID as the parent directory.

B — **For the system-specific /tmp file systems:** Each system's /tmp file system is unmounted when the system is removed from the sysplex. Thus, as we IPLed each system, it mounted the /tmp file system and successfully processed the MKDIR keyword to create the /&SYSNAME./tmp/mkdirtmpv1r5 directory.

C — **For the /pet3 file system:** When we had the /pet3 file system remotely mounted in the sysplex, the MKDIR keyword did not process on that file system. When we unmounted the /pet3 file system, since this file system is defined in the common BPXPRM00 member, the next system to initialize z/OS UNIX mounted the /pet3 file system and successfully processed the MKDIR keyword to create the /pet3/mkdirpet3v1r5 directory.

All processing messages were written to the system log, not to the console.

We also tested to make sure that multiple MKDIR keywords worked properly on the ROOT and MOUNT statements. We added the following MKDIR keywords to our BPXPRM00 member (following the MKDIR keyword that we had previously added):

```

ROOT FILESYSTEM('OMVSSPN.SYSPLEX.ROOT.FS') TYPE(HFS)
  MODE(RDWR) MKDIR('mkdirrootv1r5')
               MKDIR('mkdirrootv1r5/mkdirrootv1r5dir2')
               MKDIR('mkdirrootv1r52')
               MKDIR('mkdirrootv1r52/mkdirrootv1r52dir2')

MOUNT FILESYSTEM('OMVSSPN.&SYSNAME..TMP.FS') TYPE(HFS)
  MODE(RDWR) MOUNTPPOINT('/&SYSNAME./tmp') UNMOUNT
  PARM('FSFULL(90,5)') MKDIR('mkdirtmpv1r5')
                       MKDIR('mkdirtmpv1r5/mkdirtmpv1r5dir2')
                       MKDIR('mkdirtmpv1r52')
                       MKDIR('mkdirtmpv1r52/mkdirtmpv1r52dir2')

MOUNT FILESYSTEM('OMVSSPN.PET3.ZFS.FS') TYPE(ZFS) MODE(RDWR)
  MOUNTPPOINT('/pet3') AUTOMOVE(I,Z0,Z1,Z2,Z3) MKDIR('mkdirpet3v1r5')
               MKDIR('mkdirpet3v1r5/mkdirpet3v1r5dir2')
               MKDIR('mkdirpet3v1r52')
               MKDIR('mkdirpet3v1r52/mkdirpet3v1r52dir2')

```

We added three new MKDIR keywords to each of the above file system mounts. Using the first one (ROOT) as an example, we added MKDIR keywords, as follows:

- A** — To create a new directory under an existing directory
- B** — To create a new directory upon the next mount activity
- C** — To create a new directory under the directory created in **B**

The file systems were mounted and the directories were successfully created in the same manner as previously described. Further, we did notice that when a mount contains a series of MKDIR keywords, if one of the MKDIRs in the series fails, it does not prevent the subsequent MKDIRs from being attempted.

Testing the SYNTAXCHECK keyword

After making the above changes to the BPXPRM00 parmlib member, we issued the following command:

```
SETOMVS SYNTAXCHECK=(00)
```

Error messages, if any, are only written to the system log.

When we ran this command on a z/OS V1R5 system, it reported no errors for the MKDIR keyword. When we ran it on a z/OS V1R4 system and specified our V1R5 BPXPRMxx member, it reported errors for the MKDIR keyword.

Early in our testing, we experienced a problem such that if the syntax check encountered an uncataloged file system data set, it would then flag all file system data sets after it as being uncataloged, whether they were or were not. z/OS UNIX APAR OA05966 resolved this problem.

Temporary file system (TFS) enhancements

The temporary file system (TFS) is an in-memory physical file system that supports in-storage mountable file systems. A TFS can run in the z/OS UNIX kernel address space but, for 64-bit exploitation, it is preferable to run it in a logical file system (LFS) colony address space.

Overview of the TFS enhancements that we tested

Other enhancements to TFS in z/OS V1R5 include the following:

- “New parameters for mounting a TFS”
- “STOP and MODIFY command support for TFS colony address spaces” on page 142
- “Access control list (ACL) support” on page 143

New parameters for mounting a TFS: Each TFS mount in 64-bit mode consumes space, as requested, above the 2G bar. In addition, TFS allocates some control blocks and a buffer cache below the bar. The default buffer cache size is 1M bytes.

The **mount** command for a TFS supports the following parameters:

Parameter	Description												
-s <i>size</i>	<i>size</i> is the number of megabytes for the file system (default=1). If the specified value is larger than the size that can be supported, the maximum size will be used.												
-b <i>block</i>	<i>block</i> specifies the blocking factor used to set the size of a TFS block. Range is 0-4 (default=0). The blocking factor relates to the TFS block size as follows:												
	<table border="0"> <thead> <tr> <th style="text-align: left;">Blocking factor</th> <th style="text-align: left;">Resulting TFS block size</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">4K</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">8K</td> </tr> <tr> <td style="text-align: center;">2</td> <td style="text-align: center;">16K</td> </tr> <tr> <td style="text-align: center;">3</td> <td style="text-align: center;">32K</td> </tr> <tr> <td style="text-align: center;">4</td> <td style="text-align: center;">64K</td> </tr> </tbody> </table>	Blocking factor	Resulting TFS block size	0	4K	1	8K	2	16K	3	32K	4	64K
Blocking factor	Resulting TFS block size												
0	4K												
1	8K												
2	16K												
3	32K												
4	64K												
-c <i>cache</i>	<i>cache</i> is the amount of buffer storage, in megabytes, that TFS will use in the 31-bit address range to support a 64-bit file system. This parameter is ignored for file systems allocated in the 31-bit range. Range is 1-64; the default value is calculated based on the TFS block size such that the number of cache buffers is 256. Out-of-range values will be set to the closest range boundary.												

- u** *uid* *uid* is the numeric UID to be assigned to the file system's root directory (default=0).
- g** *group* *group* is the numeric GID to be assigned to the file system's root directory (default=0).
- p** *perm* *perm* is the permission bits, in octal, to be assigned to the file system's root directory (default=0777).
- 3** Specifies that TFS is to allocate the file system in 31-bit storage, regardless of system capabilities.

TFS dynamically determines if 64-bit addressing is enabled and, if so, places file systems above the bar. The -3 parameter on the mount command forces TFS to place a file system in 31-bit storage even if 64-bit addressing is enabled. Because locating a file system above the bar still requires a buffer pool to reside below the bar, TFS will only locate a file system above the bar when it is at least 5M bytes large and the amount of below-the-bar storage needed is less than the size of the file system above the bar.

The maximum file size that TFS can support is a function of the TFS block size. The following are the approximate maximum file sizes based on the blocking factor:

Blocking factor	Approximate maximum file size
0	2 gigabytes (G)
1	25G
2	240G
3	2 terabytes (T)
4	17T

The maximum file system size is also a function of the TFS block size, but is always limited to a maximum of $2^{31}-1$ (X'7FFFFFFF') blocks. Using the default block size of 4K, this yields a maximum file system size of about 2^{43} bytes. Increasing the block size to 64K yields a maximum file system size of about 2^{47} bytes.

For information about our test experiences, see "Testing TFS colony startup and mounting the file system" on page 143.

STOP and MODIFY command support for TFS colony address spaces: TFS, running in a colony address space, will now respond to the MVS STOP and MODIFY commands. (The STOP and MODIFY commands are not supported when TFS runs in the kernel address space.) When you issue the STOP command, TFS will stop if no file systems are mounted. The following MODIFY commands are also available to stop TFS or to force it to stop or terminate even if file systems are mounted:

MODIFY parameter	Description
STOP	This is the same as the STOP command. It causes TFS to exit if no file systems are mounted. A WTOR message is issued allowing TFS to be restarted.
TERM	Causes TFS to exit if no file systems are mounted and does not

issue a WTOR to restart TFS. The SETOMVS RESET=(xx) command can be used to start another TFS.

FORCESTOP Similar to STOP, but TFS will terminate even if there are mounted file systems.

FORCETERM Similar to TERM, but TFS will terminate even if there are mounted file systems.

For information about our test experiences, see “Testing TFS colony STOP and MODIFY commands” on page 144.

Access control list (ACL) support: TFS now supports ACLs. There are no unique external interfaces other than ACL limits. The number of ACL entries that TFS supports is limited by the block size. Each ACL uses one TFS block. For example, if the TFS block size is set to 4K (the default: -b0), it will limit the number of entries in any ACL to about 500 entries.

For information about our test experiences, see “Testing TFS colony access control list support” on page 145.

Testing the TFS enhancements

The following are some of our experiences with testing the TFS enhancements in z/OS V1R5:

Testing TFS colony startup and mounting the file system: We did the following to start TFS in a colony address space and mount the file system:

1. Created the following TFS startup procedure in *hlq.PROCLIB(TFS)*:

```
//TFS    PROC  REGSIZE=0M
//TFSGO  EXEC  PGM=BPXVCLNY,REGION=&REGSIZE,TIME=1440
//SYSIN  DD  DUMMY
//SYSRPT DD  DUMMY
//SYSOUT DD  DUMMY
//CEEDUMP DD DUMMY
//*      PEND
```

2. Modified *hlq.PARMLIB(BPXPRM00)* (our common z/OS UNIX parameter member) to specify that the TFS file system is to start in a colony address space using the TFS start up procedure:

```
FILESYSTYPE TYPE(TFS)
ENTRYPOINT(BPXTFS)
ASNAME(TFS, 'SUB=MSTR')
```

Note: We chose to start the TFS physical file system outside of JES (SUB=MSTR) which imposes some restrictions for SYSOUT (see *z/OS UNIX System Services Planning* for more information). Otherwise, to start TFS under JES, the ASNAME parameter would simply be ASNAME(TFS).

We currently do not use a TFS for the /tmp file system. We have a /tmp/tfs directory in the /tmp file system on one of our 64-bit systems (Z0) and we mount a TFS at this mount point.

3. Mounted the TFS file system in *hlq.PARMLIB(BPXPRMZ0)*, which is a system-specific z/OS UNIX parameter member (we use both BPXPRM00 and BPXPRMZ0 to initialize OMVS on system Z0):

```
MOUNT FILESYSTEM('/Z0/TMP/TFS') TYPE(TFS) MODE(RDWR)
MOUNTPOINT('/tmp/tfs') PARM('-s 10') UNMOUNT
```

Following the next IPL of system Z0, the TFS colony started in 64-bit mode outside of JES. The TFS was successfully mounted at the /tmp/tfs mount point.

4. We tested cancelling TFS and restarting it. We issued the following command to cancel TFS:

```
CANCEL TFS
```

Result: The following messages appeared:

```
BPXF063I FILE SYSTEM /Z0/TMP/TFS 121
WAS SUCCESSFULLY UNMOUNTED.
*nnnn BPXF032D FILESYSTYPE TFS TERMINATED. REPLY 'R' WHEN READY TO
RESTART. REPLY 'I' TO IGNORE.
```

We replied R to the WTOR message and TFS successfully restarted.

5. We tested unmounting and remounting the TFS file system using various combinations of parameters on the **mount** command. We also tested the -3 parameter, which forced the TFS to be mounted in 31-bit mode, even though it was running on a 64-bit system. The messages associated with a TFS mount go to the syslog.

Example: The following are examples of the TFS mount messages that appear in the syslog:

```
BPXTF006I TFS MOUNTED /Z0/TMP/TFS
BPXTF007I FILESYSTEM SIZE=1,048,576 MAX FILE SIZE=2,147,483,648
```

During our testing, we also observed the following:

- Double messages appeared in the syslog for the TFS mounts. This was resolved by z/OS UNIX APAR OA05417.
- When we tried using some incorrect parameters or out-of-range parameter values on the mount, certain displays (such as the **df** shell command and the D OMVS,F system command) still showed the incorrect information without indicating any error. In the case of an out-of-range parameter value, the parameter's default value was automatically used in place of the out-of-range value. This behavior is documented in z/OS UNIX documentation APAR OA06175.

Testing TFS colony STOP and MODIFY commands: We tested various combinations of the STOP and MODIFY commands on an active TFS colony.

1. Using the STOP TFS or MODIFY TFS,STOP command while no file system was mounted, TFS stopped, issued the WTOR message to restart, and did not allow any TFS mounts in the interim.

Example: STOP TFS or MODIFY TFS,STOP

Result:

```
*nnnn BPXF032D FILESYSTYPE TFS TERMINATED.
REPLY 'R' WHEN READY TO RESTART. REPLY 'I' TO
IGNORE.
```

When we attempted to mount the file system, we received the following error, as expected:

```
RETURN CODE 0000007A, REASON CODE 052C00B6. THE MOUNT FAILED FOR FILE SYSTEM /Z0/TMP/TFS.
```

We then replied R to the WTOR message. TFS restarted and we successfully mounted the file system.

-
2. Using the STOP TFS or MODIFY TFS,STOP command while a file system was mounted, TFS did not stop until the file system was unmounted (or FORCESTOP was issued, as below).

Example: STOP TFS or MODIFY TFS,STOP

Result:

BPXTF002I TFS TERMINATION REQUEST FAILED DUE TO ACTIVE MOUNTS

Once we unmounted the file system, the STOP and MODIFY TFS,STOP commands functioned as in case 1, above.

Note: When we issued the MODIFY TFS,FORCESTOP command with and without a file system mounted, TFS did stop each time.

-
3. Using the MODIFY TFS,TERM command while a file system was mounted, TFS did not stop until the file system was unmounted.

Example: MODIFY TFS,TERM

Result:

BPXTF002I TFS TERMINATION REQUEST FAILED DUE TO ACTIVE MOUNTS

When we issued the same command with no active file system mounts, TFS was terminated and did not issue a WTOR message to restart:

BPXTF001I TFS TERMINATION REQUEST ACCEPTED

To restart TFS, we issued the SETOMVS RESET=(00) command. TFS successfully started and we successfully mounted the file system.

-
4. Using the MODIFY TFS,FORCETERM command (both with and without a file system mounted), TFS was terminated and did not issue a WTOR message to restart.

Example: MODIFY TFS,FORCETERM

Result:

BPXTF003I TFS UNCONDITIONAL TERMINATION REQUEST ACCEPTED

To restart TFS, we issued the SETOMVS RESET=(00) command. TFS successfully started and we successfully mounted the file system.

Testing TFS colony access control list support: Support for access control lists in TFS uses the same interfaces as for the HFS and zFS file systems. We tested the **setfacl** and **getfacl** commands using files and directories in the TFS and they successfully functioned the same way they do for HFS and zFS.

z/OS UNIX enhancements in z/OS V1R6

z/OS UNIX made several enhancements in z/OS V1R6. In this section, we cover the following topics:

- “Using multipliers with BPXPRMxx parameters” on page 146
- “Using the superkill option” on page 146
- “Using wildcard characters in the automove system list (SYSLIST)” on page 148
- “Using the clear and uptime shell commands” on page 149
- “Enhanced latch contention detection” on page 150

- “Shells and utilities support for 64-bit virtual addressing” on page 151
- “Using distributed BRLM” on page 159
- “Using ISHELL enhancements” on page 161

We used the information in *z/OS UNIX System Services Planning* to help us plan and implement these enhancements.

Using multipliers with BPXPRMxx parameters

The z/OS UNIX parmlib member, BPXPRMxx, now allows the use of multipliers with certain parameters. For example, if you currently use MAXFILESIZE(1073741824); this function will let you to enter MAXFILESIZE(1M).

The BPXPRMxx parmlib parameters that accept the multiplier function are:

- MAXFILESIZE
- MAXCORESIZE
- MAXASSIZE
- MAXMAPAREA
- MAXSHAREPAGES
- IPCSHMMPAGES

Each parameter has specific limits, which are found in *z/OS MVS Initialization and Tuning Reference*, SA22-7592.

The following character abbreviations are used:

Table 14. Character Parameter Limit Multipliers

Denomination value	Character abbreviation	Bytes
Null		1
Kilo	K	1,024
Mega	M	1,048,576
Giga	G	1,073,741,824
Tera	T	1,099,511,627,776
Peta	P	1,125,899,906,842,624

Testing the multipliers

We used the SETOMVS command to test the new multiplier function. The following commands were issued:

- SETOMVS MAXSHAREPAGES=3M
- SETOMVS MAXMMAPAREA=130K
- SETOMVS MAXCORESIZE=8M
- SETOMVS MAXCORESIZE=100M
- SETOMVS IPCSHMMPAGES=16283G
- SETOMVS MAXFILESIZE=1G

Using the superkill option

The superkill option allows you to force the ending of a process or job. The superkill option is available in the following environments:

- kill command

The superkill option was added to the kill command (-K).

You must issue a kill signal to the process you want to superkill first.

Examples:

1. We issued superkill without issuing kill first.

```
kill -K 84017224
kill: FSUMF344 84017224: Cannot superkill without prior KILL signal to process
```

2. We attempted unsuccessfully to superkill all processes.

```
kill -K -- -1
kill: FSUMF342-1: Cannot superkill pid-1 (all processes)
```

3. We attempted unsuccessfully to superkill a process group.

```
kill -K -- -2
kill: FSUMF343-2: Cannot superkill a process group
```

See *z/OS UNIX System Services Command Reference*.

- console support

The superkill option was added to the MODIFY BPXOINIT console command (SUPERKILL=pid). You are not required to send a KILL signal to the process before issuing the superkill from the console support.

Example: To obtain the Process Identification (PID) of the process you want to work with, issue a command such as D 0MVS,A=ALL or (in the shell) ps -ef.

```
F BPXOINIT,SUPERKILL=50462791
BPXM027I COMMAND ACCEPTED.
IEA989I SLIP TRAP ID=X422 MATCHED. JOBNAME=ALEASE2, ASID=0171.
  <<Note that we have the S422 dump suppressed, otherwise it would have
    produced a dump for a X'422' abend, reason code X'0109'>>
BPXB0181 THREAD 2675698000000002, IN PROCESS 50462791, ENDED 153
WITHOUT BEING UNDEBLED WITH COMPLETION CODE OF 03422000, AND REASON CODE
0D2C0109.
```

See *z/OS MVS System Commands*.

- ISHELL support

There is TSO ISHELL support for superkill.

Logon to TSO → ISPF → ISH → Tools → 1. Work with processes (ps) →

After a list of processes is displayed, and you enter an "s" (S=Signal) action code, the superkill signal number will be "99".

Note that you must enter a sigkill (signal number 9), at least 3 seconds prior to entering the superkill, or you will be presented with a message with return code of Errno=79x (the parameter is incorrect), and Reason=0D1005D8 (JRSigkillNotSent).

The following set of restrictions applies:

- You cannot do a superkill via pthread_kill or sigqueue(). The superkill option is ignored for these services.
- You cannot do a superkill to a group or specify a PID of -1 (kill everyone).
- Superkills will be deferred in the case where a target process has blocked all signals via the BPX1SDD service. The 'defer signal' function was created for conditions which cannot deal with USS abends on their system task. For this reason superkills will also have no effect on these processes. The kill() will not fail but will simply be ignored by the target process.
- A regular SIGKILL must be sent to a process before it can be superkilled. If not, the attempt will result in EINVAL/JRSigkillNotSent. This is analogous to the required 'cancel' before a 'force arm'. However, if using the console form of superkill, the sigkill (cancel) before does not apply.

If the environment is valid then the target process will be abended with a X'422' abend, reason code X'0109'.

Using wildcard characters in the automove system list (SYSLIST)

We tested the new wildcard character support for the automove system list (SYSLIST) on our systems. This function lets you use wildcards in certain situations when you specify the automove system list. Before this enhancement, if an AUTOMOVE INCLUDE with the SYSLIST function was used for a file system mount, only the list of systems specified were eligible to become owners of the file system. If none of the systems specified were active, the system would then unmount the file system. Now, however, using the wildcard support, you can both specify your preferred ownership system candidate systems and use a wildcard to allow any of the remaining systems in the shared HFS sysplex eligible to become the file system owner to prevent the system from unmounting the file system.

We tested this support by mounting a file system using the INCLUDE statement, and specifying a subset of the systems in our 14-way sysplex as well as a wildcard character (*). In the example below shows a MOUNT statement for filesystem OMVSSPN.PET1.FS with an AUTOMOVE system list:

```
MOUNT FILESYSTEM('OMVSSPN.PET1.FS') TYPE(HFS) MODE(RDWR)
      MOUNTPOINT('/pet1') AUTOMOVE(I,Z1,Z2,Z3,*)
```

Display of mounted filesystem showing automove attributes follows:

```
HFS          15 ACTIVE                      RDWR
  NAME=OMVSSPN.PET1.FS
  PATH=/pet1
  OWNER=J80      AUTOMOVE=I CLIENT=Y
  INCLUDE SYSTEM LIST:  Z1          Z2          Z3          *
```

Note that system J80 is the owning system for the file system. In the event that either someone issues a file system shutdown on system J80 or if system J80 leaves the sysplex, the system transfers ownership of the file system to system Z1, if it's active. If Z1 is inactive, Z2 gets ownership, and so forth down the list. If system Z3 isn't active then any remaining active image in the sysplex will be randomly selected to become the owner.

You can also use either AUTOMOVE(I,*) or AUTOMOVE(YES) to specify that **any** active system in the sysplex is eligible to take ownership of the file system. For example, the following two MOUNT statements would yield the same result that any active system can take over OMVSSPN.PET5.ZFS:

```
1
MOUNT FILESYSTEM('OMVSSPN.PET5.ZFS') TYPE(ZFS) MODE(RDWR)
      MOUNTPOINT('/pet5') AUTOMOVE(I,*)
2
MOUNT FILESYSTEM('OMVSSPN.PET5.ZFS') TYPE(ZFS) MODE(RDWR)
      MOUNTPOINT('/pet5') AUTOMOVE(YES)
```

In all methods of issuing a MOUNT, the wildcard in AUTOMOVE must **always** be the last item in the system list. This includes PARMLIB, TSO, shell, ishell, C program, assembler program and REXX program methods.

Note that the system does not validate the system identifiers specified in the system list (Z1, for example) until ownership of a file system is being transferred. However, using the wildcard at the end of the list ensures that the file system can always find a new owner even if you spell every single system identifier incorrectly. This feature will prevent your file system from being unmounted unexpectedly. The wildcard feature is very useful for installations that want to:

- Make sure a file system is always mounted
- Do not want to list every system in the system list

Using the clear and uptime shell commands

We tested the new V1R6 clear and uptime shell commands:

- “Using the clear command”
- “Using the uptime command”

For more information about these commands, see the *z/OS UNIX System Services Command Reference*.

Using the clear command

The clear shell command clears the screen of all previous output and places the prompt at the top of the page. Similarly to the `tput clear` command, you cannot use this command if you've accessed the shell using a 3270 window. The clear utility has no options, and if you try to enter any options, the system issues a usage message.

The following is our experience with the clear utility:

- **From OMVS (TSO):** From the shell, we entered `clear`. The command cleared the screen, leaving a blank line and the prompt at the top of the screen. We were able to use the PF7 key to return to previous pages. We also entered more display commands, `ls -al` for example, and then entered the clear command again. Again, the command cleared the screen, put the prompt on the top. We used PF7 (BACKSCR) and PF8(SCROLL) to navigate.
- **From rlogin or telnet (AIX):** We accessed from AIX using the `rlogin` command and then entered `clear`. The command cleared the screen and the prompt was displayed at the top of the screen. We also entered more display commands, `ls -al` for example, and then entered the clear command again. Again, the command cleared the screen, put the prompt on the top. Note that from `rlogin`, the clear command actually removes the data on the screen, so that even if you scroll back, the data will no longer be displayed. When we did scroll back, the command prompt went back to the top of the screen when we began typing again.
- **From telnet (3270 window):** We accessed the shell using telnet on TSO then entered `clear`. The command **did not** clear the screen. The prompt is simply redisplayed.

Using the uptime command

The uptime command gives a one-line display with the following information:

- Current time
- How long the system has been running
- Number of users who are currently logged into z/OS UNIX and the system load averages for the past 1, 5, and 15 minutes. Load averages are not supported on z/OS UNIX, and are displayed as 0.00

We tested the uptime command from telnet, rlogin, and TSO (OMVS) sessions. Note that the uptime command has no documented parameters, and when we tried to use parameters, we received a usage message.

The following example shows how we entered the uptime command and the output from it:

```
# uptime
06:49PM up 5 day(s), 01:55, 1 users, load average: 0.00, 0.00, 0.00
```

The load average value is always 0.00 on our systems because we are running on z/OS UNIX.

Enhanced latch contention detection

In z/OS V1R6 changes were implemented to eliminate outages caused by address spaces that terminate without cleaning up the ownership of the acquired USS global resource serialization latch. The kernel now detects latch contention on a timed basis and initiates one of the following:

- Attempts to correct the problem
- Issues a message that contention exists.

An action is taken if the latch that is causing contention is held for an excessive amount of time. If the kernel detects that the oldest latch holder's address space or process no longer exist, it corrects the latch contention problem. If the kernel detects a latch holder's address space and process still exist, the following eventual action message is displayed indicating that additional actions might be required:

BPXM056E UNIX SYSTEM SERVICES LATCH CONTENTION DETECTED

If the message does not get DOMed after a reasonable amount of time, your installation should have an automation script or an operator to react to this message and eventually issue the **F BPXOINIT,RECOVER=LATCHES** command to attempt to resolve the contention.

The abnormal termination is caused by a 422-1A5 abend that generates a system dump. The dump is generated because of the indication of an internal system problem. If more than one latch is in contention, multiple tasks can abend and result in multiple dumps being requested. However, prior to issuing the recovery command, it is advisable to use the D GRS command to determine the resources and address spaces that are involved with the contention. For example, use D GRS,C.

Once the latch contention is resolved, message BPXM056E will be DOMed. This command can take several minutes to resolve the latch contention. If the command cannot resolve the latch contention within a reasonable amount of time, the following eventual action message is displayed and message BPXM056E is DOMed: *BPXM057E UNIX SYSTEM SERVICES LATCH CONTENTION NOT RESOLVING.*

Note: A new 422-1A5 abend is introduced that can terminate a user task holding a USS latch for an excessive amount of time.

The following new message indicates the entry of an unsupported operand when using the **F BPXOINIT,RECOVER=** command: *BPXM058I MODIFY BPXOINIT RECOVER COMMAND REJECTED.*

Testing contention recovery

The following example shows what occurred when we used **F BPXOINIT,RECOVER=LATCHES**:

The SYS.BPX.AP00.PRTB1.PPRA.LSN type latches were held for approximately 6 hours on one system. After issuing **F BPXOINIT,RECOVER=LATCHES**, these latches were relieved (received abend 422/1A5), and the BPXM056E message was DOMed. In our installation no dump was taken because we have S422 abends suppressed (SLIP SET,C=422,ID=Y422,A=NODUMP,E).

Example:

```

D GRS,C
ISG343I 10.34.16 GRS STATUS 167
S=STEP   SYSZBPX  PROCINIT
SYSNAME  JOBNAME          ASID      TCBADDR  EXC/SHR  STATUS
JC0      U078023          0242      008E79C0 EXCLUSIVE OWN
JC0      U078023          0242      008E7B58 EXCLUSIVE WAIT
NO REQUESTS PENDING FOR ISGLOCK STRUCTURE
LATCH SET NAME:  SYS.BPX.AP00.PRTB1.PPRA.LSN
CREATOR JOBNAME:  OMVS      CREATOR ASID: 000F
  LATCH NUMBER:  908
    REQUESTOR  ASID  EXC/SHR  OWN/WAIT
    U078025    0221  EXCLUSIVE  OWN
    U078023    0242  EXCLUSIVE  WAIT

```

```

F BPXOINIT,RECOVER=LATCHES
BPXM027I COMMAND ACCEPTED.
IEA989I SLIP TRAP ID=Y422 MATCHED.  JOBNAME=U078025 , ASID=0221.
BPXP018I THREAD 26A6D5F000000002, IN PROCESS 524404, ENDED 171
WITHOUT BEING UNDUBBED WITH COMPLETION CODE 0F422000, AND REASON CODE
000001A5.
BPXM067I UNIX SYSTEM SERVICES LATCH CONTENTION RESOLVED

```

```

D GRS,C
ISG343I 10.36.39 GRS STATUS 188
NO ENQ RESOURCE CONTENTION EXISTS
NO REQUESTS PENDING FOR ISGLOCK STRUCTURE
NO LATCH CONTENTION EXISTS

```

There were instances when we issued the **F BPXOINIT,RECOVER=LATCHES** and received the BPXM067I message with no S422/1A5 abends. This indicates that there is no latch contention for this function to attempt recovery from.

Example:

```

F BPXOINIT,RECOVER=LATCHES
BPXM027I COMMAND ACCEPTED.
BPXM067I UNIX SYSTEM SERVICES LATCH CONTENTION RESOLVED

```

Shells and utilities support for 64-bit virtual addressing

We tested several z/OS UNIX shell utilities that were enhanced to support 64-bit virtual addressing. These enhanced utilities support the creation and use of 64-bit applications.

Overview of 64-bit support

The utilities that we tested fall into one of three groups:

- Utilities that report information about an executable or object file that has been compiled in 64-bit mode (such as **nm** and **file**): These utilities recognize 64-bit executables and symbols within executables and object files and libraries.
For example, the **file** utility indicates whether a file is an executable and, if so, whether it is a 64-bit executable.
- Utilities that report information about 64-bit processes (such as **ps** and **ipcs**): These utilities report information about memory usage by 64-bit processes and threads.
For example, **ps** reports memory usage above the bar by a running process. The **ipcs** utility reports information on shared memory that can be allocated and attached above the bar by 64-bit programs.
- Utilities that manage object files or executables (such as **ar**, **cp** and **mv**, and the **lex** and **yacc** libraries). These utilities support 64-bit executables and object files.
For example, the **ar** utility, which creates and manages object libraries, now stores the addressing mode symbols. The **cp** and **mv** utilities, when used to

move executables from MVS data sets to the z/OS UNIX file system, re-binds 64-bit executables. Also, **lex** and **yacc** provide object libraries with 64-bit versions of functions to support 64-bit code created using **lex** or **yacc**.

The shell utilities themselves are not compiled as 64-bit applications.

Examples of the utilities that we tested

The following are some examples that highlight our testing of the 64-bit virtual addressing support in various shell utilities.

The file utility: The **file** utility determines the format of each file by inspecting the attributes and, for a regular file, by reading an initial part of the file. If the **file** is an executable, file determines its addressing mode for output. If the file is not an executable, **file** compares it to templates found in a magic file to determine the file type.

A word about the magic file: The magic file defines the following output text for an executable:

```
OpenEdition MVS executable
```

This wording is outdated; therefore, you may wish to update your `/etc/magic` file using the new `/samples/magic` file. The updated `/samples/magic` file now defines the following output text for an executable:

```
z/OS UNIX executable
```

Examples of testing the file utility: The following examples highlight our experiences testing the **file** utility.

Example: `file /bin/makedepend`

Result: The file utility does not support the display of information for external links and issues the following response:

```
FSUM8718 /bin/makedepend: cannot open: EDC5129I No such file or directory.
```

The output from the **ls -al** command shows that `/bin/makedepend` is an external link:

```
erwxrwxrwx 1 ALEASE1 OMVSGRP 8 Jun 22 09:01 /bin/makedepend -> CCNEMDEP
```

Example: `file /bin/*`

Result: The following excerpt of the command response shows both AMODE 31 and AMODE 64 executables:

```
⋮  
/bin/dbx24:.z/OS Unix executable  
/bin/dbx31:.z/OS Unix executable  
/bin/dbx31vdbg:.z/OS Unix executable (amode=31)  
/bin/dbx64:.z/OS Unix executable  
/bin/dbx64vdbg:.z/OS Unix executable (amode=64)  
/bin/dce_err:.z/OS Unix executable (amode=31)  
/bin/dce_login:.z/OS Unix executable (amode=31)  
/bin/dcecf_postproc:.commands text - Bourne or POSIX shell script  
⋮
```

The nm utility: The man pages for the **nm** utility indicate that the `-M` option inserts three columns preceding the symbol name in the command output. The three columns have the following format:

```
rmode amode compiler_options
```

The rmode and amode columns display one of the following:

```
24-bit mode
31-bit mode
64-bit mode
ANY mode
MIN mode
Undetermined or not applicable
```

The compiler options column displays a character for each compiler option in effect, or a dash if no options are in effect, as follows:

```
I Symbol is compiled with IPA. (IPA will not be seen when running nm against
an executable as that information is no longer available.)
X Symbol is compiled with XPlink.
```

Examples of testing the nm utility: The following examples highlight our experiences testing the **nm** utility.

As in the example of the **file** utility above, the **nm** utility does not support the display of information for external links and issues the same error message as above.

Example: `nm -M /bin/dbx64vdbg`

Result:

```
560 T 64 64 - BPX4PTR
560 U --- --- - BPX4PTR
224 T MIN MIN X BPXISD64
208 T 64 64 X BPXISD64#C
0 U --- --- - B_IMPEXP
0 U --- --- - B_LIT
0 U --- --- - B_PRV
208 U --- --- - B_TEXT
0 U --- --- - B_TEXT
584 U --- --- - B_TEXT
760 U --- --- - B_TEXT
560 U --- --- - B_TEXT
952 U --- --- - B_TEXT
840 U --- --- - B_TEXT
608 T MIN MIN - CEELLIST
760 T 64 64 - CELQETBL
760 U --- --- - CELQETBL
584 U --- --- - CELQLLST
584 U --- --- - CELQLLST
584 T 64 64 - CELQLLST
0 T 64 64 - CELQSTRT
0 U --- --- - CELQSTRT
960 T MIN MIN X CELQTLOC
952 T 64 64 - CELQTLOE
840 T 64 64 - CELQTRM
840 U --- --- - CELQTRM
224 T MIN MIN X CELQVDBG
```

```

|           0 U --- --- -   C_DATA64
|           0 T ANY ANY -   IEWBCIE
|           0 T ANY ANY -   IEWBLIT
|           0 U --- --- -   IEWBLIT

```

The ps utility: The **ps** utility now shows the memlimit and amount of storage in use above the 2-gigabyte bar. Two new output specifiers (vszlm64 and vsz64) are also available.

The following are the pertinent fields of the **ps** display:

vsz	Displays the amount of memory (virtual storage) that the process is using, as a decimal number of kilobytes.
vszlm64	Displays the maximum amount of virtual storage above the 2-gigabyte bar allowed for the current process. In the display, each value is followed by a multiplier indicating the units represented: (space) = no multiplier; K = Kilo; M = Mega; G = Giga; T = Tera; P = Peta.
vsz64	Displays the virtual storage used above the 2-gigabyte bar. In the display, each value is followed by a multiplier indicating the units represented: (space) = no multiplier; K = Kilo; M = Mega; G = Giga; T = Tera; P = Peta.

Example of testing the ps utility: The following example highlights our experiences testing the **ps** utility.

1. We issued the following command to create a 64-bit compiled module for a private program called brlmcntl.c using the new xlc compile invocation utility available in z/OS V1R6. (We used the default configuration file at /usr/lpp/cbclib/xlc/etc/xlc.cfg and the utility command at /usr/lpp/cbclib/xlc/bin.)

```

/usr/lpp/cbclib/xlc/bin/xlc_64 -o brlmcntl.out64 brlmcntl.c

```

This creates an executable named brlmcntl.out64.

2. Our first attempt to run the new program failed because we had used the default MEMLIMIT value, which is zero. The following is an example of how we invoked the program and the resulting error message:

```
brlmcntl.out64 brlmf1 120
```

Result:

```

1 + Done(137) brlmcntl.out64 /
17695742      Killed ./brlmcntl.out64

```

3. To resolve the problem in step 2, we issued the following MVS operator command to set the MEMLIMIT for the process under which we were running the program. In this case, that process was the shell session. We set the MEMLIMIT to 10M.

```
SETOMVS PID=68026746,MEMLIMIT=10M
```

To verify this change, we issued the following operator command and checked the MEMLIMIT value:

```
DISPLAY OMVS,L,PID=068026746
```

Note: When setting the MEMLIMIT on your systems, carefully consider factors such as a system-wide setting, override capability, and process-specific settings.

4. Our brlmcntl program takes as the second parameter the number of seconds to remain active. We forked three programs to hold for two minutes (120 seconds) each, as follows:

```
brlmcntl.out64 brlmf1 120 &
brlmcntl.out64 brlmf2 120 &
brlmcntl.out64 brlmf3 120 &
```

5. We issued the ps command to display the vsz64 and vsz1mt64 fields, as follows:

```
ps -e -f -o jobname,pid,ppid,vsz,vsz64,vsz1mt64,args
```

Result:

JOBNAME	PID	PPID	VSZ	VSZ64	VSZLMT64	COMMAND
...						
ALEASE19	68026754	917883	4804	8388608	10M	brlmcntl.out64 brlmf1 120
ALEASE11	917891	917883	4804	8388608	10M	brlmcntl.out64 brlmf2 120
ALEASE12	917892	917883	4804	8388608	10M	brlmcntl.out64 brlmf3 120
...						

The above display indicates that the programs are using 8M of storage above the bar (VSZ64) and that the limit is 10M (VSZLMT64).

The ipcs utility: The **ipcs** utility writes information to the standard output stream about active inter-process communication facilities. The PGSZ and SEGSZ fields are added to the list of fields under the **-x** option. The SEGSZPG field is now displayed with the **-x** option. These fields are defined as follows:

SEGSZPG

The size, in pages, of the associated shared memory segment.

PGSZ The page size of the associated shared memory segment.

SEGSZ

The size, in bytes, of the associated shared memory segment.

Note that the new PGSZ field is not properly defined in the z/OS V1R6 documentation or the man pages for the **ipcs** command. See documentation APAR OA08772 for more information.

Examples of testing the ipcs utility: The following examples highlight our experiences testing the **ipcs** utility.

We ran the command using the three options (**-a**, **-b**, and **-x**) that display the SEGSZPG, PGSZ, and SEGSZ fields on a system where we had some shared memory activity. The following examples show excerpts of the displays.

Example: ipcs -a

Result:

```

:
: Shared Memory:
T   ID   KEY   MODE   OWNER   GROUP   CREATOR   CGROUP   NATTC   SEGSZPG PGSZ   SEGSZ   ATIME   DTIME   CTIME   CPID   LPID
m  40004 0x023625dc --rw-rw---- LORAINO  IMWEB  LORAINO  IMWEB   1       1       4K    1793    20:33:41 00:00:00 20:33:41 327706 327706
```

```

m      40005 0x033625dc --rw-rw---- LORAIN0 IMWEB LORAIN0 IMWEB      1      10240 4K      41943040 20:33:41 00:00:00 20:33:41      327706      327706
m      40006 0x043625dc --rw-rw---- LORAIN0 IMWEB LORAIN0 IMWEB      1      1      4K      99      20:33:41 00:00:00 20:33:41      327706      327706
:

```

Example: `ipcs -b`

Result:

```

:
Shared Memory:
T      ID      KEY      MODE      OWNER      GROUP      SEGSZPG  PGSZ      SEGSZ
m      40004 0x023625dc --rw-rw---- LORAIN0     IMWEB      1         4K        1793
m      40005 0x033625dc --rw-rw---- LORAIN0     IMWEB      10240     4K        41943040
m      40006 0x043625dc --rw-rw---- LORAIN0     IMWEB      1         4K        99
:

```

Example: `ipcs -x`

Result:

```

:
Shared Memory:
T      KEY      OWNER      GROUP      SEGSZPG  PGSZ      SEGSZ      ATPID      ATADDR      INFO
m 0x023625dc LORAIN0     IMWEB      1         4K        1793      327706     0x00000000273f6000
m 0x033625dc LORAIN0     IMWEB      10240     4K        41943040 327706     0x0000000027500000      M
m 0x043625dc LORAIN0     IMWEB      1         4K        99        327706     0x00000000273fd000
:

```

The *ar* utility: You can use the *ar* utility to store multiple versions of the same object file within one archive library. This is useful if you are providing an archive library which may be used to resolve references from code compiled with various compiler options. These options cause differences in the object files which must be matched with the archive library member attributes. Attributes for *ar* are *AMODE*, *XPLINK*, and *IPA*. The *ar* utility stores the attribute information for each entry in the symbol table. The linkage editor uses the attribute information to resolve external references with the appropriate archive library member. Because the names of archive library members consist of only the final component of the path name, the member names must be unique for the different object file versions. It's a good idea to establish a naming convention for the object files and to implement build procedures to generate the correct names.

To display the attributes of the symbols within an object file or an archive library of object files, use the *nm* command, which displays the symbol table of object, library, or executable files.

Examples of testing the ar utility: The following example highlights our experiences testing the *ar* utility.

We performed the following steps to create an archive library, add members to it, delete members from it, and display its contents:

1. Using the *xlc* utility, we compiled a source file (`displayfs.c`) using various compiler options—31-bit, 31-bit with *XPLINK*, and 64-bit (which also forces *XPLINK*)—as follows:

```

/usr/lpp/cbclib/xlc/bin/xlc -o displayfs.out31 displayfs.c
/usr/lpp/cbclib/xlc/bin/xlc_x -o displayfs.out31x displayfs.c
/usr/lpp/cbclib/xlc/bin/xlc_64 -o displayfs.out64 displayfs.c

```

2. We created an archive library called `libdisplayfs.a` containing the three members:

```

ar -ruv libdisplayfs.a displayfs.out31 displayfs.out31x displayfs.out64

```

-
3. The following command displays the contents of the archive library:

```
ar -tv libdisplayfs.a
```

Result: The archive library contains three members:

```
rw-rw-rw- 0/0 77824 Jul 28 13:17 2004 displayfs.out31
-rwxr-xr-x 0/0 81920 Jul 28 13:19 2004 displayfs.out31x
-rwxr-xr-x 0/0 61440 Jul 28 13:17 2004 displayfs.out64
```

4. We created another 64-bit compiled object (as in step 1) named `displayfs.out64b` and then added it to the archive library using the following command:

```
ar -rcv libdisplayfs.a displayfs.out64b
```

Result:

```
a - displayfs.out64b
```

5. Displayed the contents once again:

```
ar -tv libdisplayfs.a
```

Result: The archive library now contains four members:

```
rw-rw-rw- 0/0 77824 Jul 28 13:17 2004 displayfs.out31
-rwxr-xr-x 0/0 81920 Jul 28 13:19 2004 displayfs.out31x
-rwxr-xr-x 0/0 61440 Jul 28 13:17 2004 displayfs.out64
-rwxr-xr-x 0/0 61440 Jul 28 13:32 2004 displayfs.out64b
```

6. We then deleted the last member that we added (`displayfs.out64b`):

```
ar -dsv libdisplayfs.a displayfs.out64b
```

Result:

```
d - displayfs.out64b
```

7. The following command attempts to display the member we just deleted:

```
ar -tv libdisplayfs.a displayfs.out64b
```

Result:

```
ar: displayfs.out64b not found
```

8. Displayed the contents one more time:

```
ar -tv libdisplayfs.a
```

Result:

```
rw-rw-rw- 0/0 77824 Jul 28 13:17 2004 displayfs.out31
-rwxr-xr-x 0/0 81920 Jul 28 13:19 2004 displayfs.out31x
-rwxr-xr-x 0/0 61440 Jul 28 13:17 2004 displayfs.out64
```

The `cp` and `mv` utilities: As of z/OS V1R6, the `cp` and `mv` utilities can now copy or move 64-bit executables between MVS and the z/OS UNIX file systems and

correctly rebind executables, just as these utilities do for 24- and 31-bit applications. There are no external changes to the way you use these commands.

The ulimit utility: The **ulimit** utility sets or displays resource limits on processes created by the user. There are two new options available, as follows:

- A Set or display the maximum address space size for the process, in units of 1024 bytes. If the limit is exceeded, storage allocation requests and automatic stack growth will fail. An attempt to set the address space size limit lower than the size that is already in use will fail.
- M Set or display the amount of storage above the 2-gigabyte bar (MEMLIMIT), in one megabyte increments, that a process is allowed to have allocated and unhidden.

You can specify unlimited as the new limit. Using these options without specifying a limit value will simply display the current setting. These new limits also appear in the display using the `–a` option.

Examples of testing the ulimit utility: The following examples highlight our experiences testing the **nm** utility.

Example: `ulimit -a`

Result:

```
core file      15842b
cpu time      unlimited
data size     unlimited
file size     unlimited
stack size    unlimited
file descriptors 65535
address space 72808k
memory above bar 10m
```

Example: `ulimit -M`

Result:

```
10
```

Example: `ulimit -A`

Result:

```
72808
```

The limit and unlimit (tcsh) utilities: The **limit** utility limits the consumption of resources by the current process and each process it creates so that, individually, those processes cannot exceed a maximum-use value for a specified resource. If no maximum-use value is specified on the command, then the current limit is displayed. If no resource is specified, then the limitations for all resources are displayed.

The `tcsh limit` command includes two new resources, `memlimit` and `addressspace`, as follows:

`addressspace` The maximum address space size for the process, measured in kilobytes. If a process exceeds the limit, functions such as `malloc()`

and mmap() will fail. Also, automatic stack growth will fail. An attempt to set the address space size limit lower than the size that is already in use will fail.

memlimit The amount of storage, in megabytes, above the 2-gigabyte bar that a process is allowed to have allocated and unhidden at any given time.

The **unlimit** utility removes the limitation on the specified resource. If no resource is specified on the command, then all resource limitations are removed.

Examples of testing the limit and unlimit utilities: The following examples highlight our experiences testing the **limit** and **unlimit** utilities.

Example: `limit`

Result:

```
cputime          unlimited
filesize         unlimited
datasize         unlimited
stacksize        unlimited
coredumpsize     7921 kbytes
descriptors      65535
addressspace     72808 kbytes
memlimit         10 megabytes
```

Example:

```
limit memlimit 8
limit memlimit
```

Result:

```
memlimit         8 megabytes
```

Example:

```
limit memlimit 8m
limit memlimit
```

Result:

```
memlimit         8 megabytes
```

Using distributed BRLM

With V1R6, byte range lock manager (BRLM) has changed to support distributed BRLM. Instead of a single, central BRLM, distributed BRLM means that each system in the sysplex runs a separate BRLM, which is responsible for locking files in the file systems owned and mounted on that system. This means that a file system can be moved while byte range locks are held for files in the file system. When a file system changes owners, the corresponding locking history changes BRLM servers at the same time. (Note that this is not the case when a system failure occurs.)

For this reason, distributed BRLM is now the only supported method when all systems are at the V1R6 level and distributed BRLM is automatically activated on every system for an all z/OS V1R6 sysplex. Each system runs a BRLM and is responsible for handling lock requests for files whose filesystems are mounted and owned locally on that system.

If you are already running with a z/OS UNIX couple data set (CDS) indicating that distributed BRLM is enabled (DISTBRLM set to 1), there is no change required to activate distributed BRLM for V1R6. Likewise, if your sysplex only has systems at the V1R6 level, there is no change required, because distributed BRLM is the default. V1R6 systems ignore the z/OS UNIX CDS DISTBRLM setting.

However, if you migrate to V1R6 by running mixed levels in a sysplex, you should enable distributed BRLM before IPLing the V1R6 system because a V1R6 system may attempt to activate distributed BRLM when the central BRLM server leaves the sysplex, regardless of the z/OS UNIX CDS setting. The inconsistency between distributed BRLM being active and central BRLM being defined in the z/OS UNIX CDS can cause an EC6-BadOmvsCds abend on downlevel systems. This is a notification-only abend indicating that the CDS should be updated. z/OS UNIX will still operate normally, and distributed BRLM will be active in the sysplex. See *z/OS Migration* for more information.

We used the following display command on each system to display whether distributed BRLM is enabled and active:

```
F BPXOINIT,FILESYS=DISPLAY,GLOBAL
```

We got the following output from the display command:

```
BPXM027I COMMAND ACCEPTED.
BPXF040I MODIFY BPXOINIT,FILESYS PROCESSING IS COMPLETE.
BPXF041I 2004/08/10 14.10.09 MODIFY BPXOINIT,FILESYS=DISPLAY,GLOBAL
SYSTEM  LFS VERSION  ---STATUS----- RECOMMENDED ACTION
Z0      1.  6.  0 VERIFIED                NONE
JA0     1.  6.  0 VERIFIED                NONE
TPN     1.  6.  0 VERIFIED                NONE
Z1      1.  6.  0 VERIFIED                NONE
J90     1.  6.  0 VERIFIED                NONE
JF0     1.  6.  0 VERIFIED                NONE
JB0     1.  6.  0 VERIFIED                NONE
JC0     1.  6.  0 VERIFIED                NONE
JE0     1.  6.  0 VERIFIED                NONE
Z2      1.  6.  0 VERIFIED                NONE
Z3      1.  6.  0 VERIFIED                NONE
J80     1.  6.  0 VERIFIED                NONE
JG0     1.  6.  0 VERIFIED                NONE
JH0     1.  6.  0 VERIFIED                NONE
CDS VERSION= 2          MIN LFS VERSION= 1.  6.  0
BRLM SERVER=N/A        DEVICE NUMBER OF LAST MOUNT= 9266
MAXIMUM MOUNT ENTRIES= 800  MOUNT ENTRIES IN USE= 699
MAXIMUM AMTRULES=      51  AMTRULES IN USE= 9
DISTBRLM ENABLED=YES   DISTBRLM ACTIVE=YES
```

We're planning to use the enhancement allowing a filesystem to be moved even when it contains locked files. Prior to z/OS V1R6, you would receive an enomove return code error results if you issued filesystem move commands while a file was open and was byte range locked. In z/OS V1R6, this error code now only occurs if you have a pre-z/OS R6 system in the sysplex. In a pre-z/OS V1R6 system, an enomove return code prevents the filesystem move. In order to move filesystems that contain locked files, all systems in the sysplex must be at the z/OS V1R6 level.

Note that the system does not necessarily report an enomove return code.

Restriction: With distributed BRLM, certain cross-system deadlock scenarios may not be detected. Locking applications must ensure that they do not cause deadlocks. See *z/OS UNIX System Services Planning*.

Using ISHELL enhancements

In z/OS R6, we tested the following enhancements to the ISPF shell (ISHELL) commands. For additional ISHELL information, use the help panels that come with the product.

Wild card support for the command filter: UNIX System Services (USS) now supports the command **filter** on the directory list. If you enter the ISHELL command without any argument, a panel will be displayed to enter the new filter enhancements. You can use any characters with the wild card character (*) in the filter, and the wild card character * can match any number of characters or no characters. For example, the command filter *.c will show only files that end with .c. Filter *a* will show only file names that contain an a. The filter is case sensitive. If there is not a match for the filter then the entire directory list is given and the filter is disabled. The following example shows a series filters issued, and the output displayed:

```
filter *.c
EUID=0 *.c /u/lates/
  Type Filename
  _ File hello.c
  _ File rlimit.c
  _ File wstatvfs.c

filter *a*
EUID=0 *a* /u/lates/
  Type Filename
  _ File a.out
  _ Dir aaa
  _ File chkosname.jar
  _ File copymap

filter w*
EUID=0 w* /u/lates/
  Type Filename
  _ File wstatvfs
  _ File wstatvfs.c
```

Command line position panel option: To test the new command line position enhancements from an ISHELL panel we selected OPTIONS/ADVANCED. This gives us 3 command line selections; top, bottom and inherit, as follows:

```
Advanced Options

Select options

  _ Bypass delete confirmations
  _ Bypass exit confirmation
  _ No auto-skip on action panels
  _ Always start initial panel with current directory

Command line position:
  _ 1. Top
  _ 2. Bottom
  _ 3. Inherit
```

We tested all three command line positions successfully - they each positioned the command line as expected.

Options panel for displaying Permissions: To test the new permissions display from the ISHELL panel, we selected the OPTIONS/DIRECTORY list. This panel lists selections for displaying the permissions listed:

Directory List Options

Selected options and fields to be displayed with /

- _ File type (4 columns)
- _ Permissions (4 columns, octal)
- _ Permissions (10 columns, rwx)
- _ Change time (16 columns)
- _ Owner (9 columns)
- _ File size (10 columns)

- _ View/change sort options...
- _ View/change file name highlighting...
- 7 Verbose directory list panel
- _ Null Enter refreshes list
- _ Stop processing multiple selections after a message

We selected "Permissions (4 columns, octal)" and displayed directory /pet6 as follows:

```
EUID=0 /pet6/
  Type Perm Filename
_ Dir 750 .
_ Dir 755 ..
_ File 644 a
_ File 644 A
_ File 644 b
_ File 644 B
_ File 644 c
_ File 644 C
_ File 644 d
_ File 644 D
_ File 644 e
_ File 644 E
_ File 644 f
```

Next we selected "Permissions (10 columns, rwx)" and displayed directory /pet6 as follows:

```
EUID=0 /pet6/
  Type Permission Filename
_ Dir rwxr-x--- .
_ Dir rwxr-xr-x ..
_ File rw-r--r-- a
_ File rw-r--r-- A
_ File rw-r--r-- b
_ File rw-r--r-- B
_ File rw-r--r-- c
_ File rw-r--r-- C
_ File rw-r--r-- d
_ File rw-r--r-- D
_ File rw-r--r-- e
_ File rw-r--r-- E
_ File rw-r--r-- f
```

We then selected **both** permission selections to get the following display:

```
EUID=0 /pet6/
  Type Perm Permission Filename
_ Dir 750 rwxr-x--- .
_ Dir 755 rwxr-xr-x ..
_ File 644 rw-r--r-- a
_ File 644 rw-r--r-- A
_ File 644 rw-r--r-- b
_ File 644 rw-r--r-- B
_ File 644 rw-r--r-- c
_ File 644 rw-r--r-- C
```

```

_ File 644 rw-r--r-- d
_ File 644 rw-r--r-- D
_ File 644 rw-r--r-- e
_ File 644 rw-r--r-- E
_ File 644 rw-r--r-- f

```

Option for preserving extended attributes on copy: To test this function, we first updated file 'a' in /pet6 to have extended attributes using the following command:

```
extattr +aps a
```

We then used the ISHELL panel to do a copy (c) of file 'a' to file 'aaaa':

```

EUID=0 /pet6/
  Type Perm Owner -----Size Filename
_ Dir 750 LORAIN0 1952 .
_ Dir 777 LORAIN0 24576 ..
c File 644 LORAIN0 0 a
_ File 644 LORAIN0 0 A
_ File 644 LORAIN0 0 b
_ File 644 LORAIN0 0 B

```

We got the following panel, where we selected **1** to copy our file into another file:

```

Copy from a File

Copying from file:
/pet6/a

Destination for copy:
1 1. File...
  2. Data set...

Select additional options for data set copy:
_ Binary copy
_ Conversion...

```

We were prompted to enter the destination file name (aaa), as follows:

```

Enter the Pathname

Change this to the pathname of the target file:          More:  +
/pet6/aaaa
_____
_____
_____

```

We were also prompted to enter the permissions for the destination file:

```

Enter File Permissions

Permissions . . 644 (3 digits, each 0-7)

```

Now we get to the extended attributes panel, where we selected to copy the extended attributes to the destination file:

```

Extended Attributes

The selected file contains extended attributes. Select the option below
to copy all of the extended attributes to the new file. Note that
authority may be needed to set some extended attributes.

```

```
s Copy extended attributes
```

When this is done, we get the following panel:

```

EUID=0 /pet6/
Type Perm Owner -----Size Filename
_ Dir 750 LORAIN0 1952 .
_ Dir 777 LORAIN0 24576 ..
_ File 644 LORAIN0 0 a
_ File 644 LORAIN0 0 A
_ File 644 LORAIN0 0 aaaa
_ File 644 LORAIN0 0 b

```

Finally, we listed the file attributes using the directory panel and confirmed the extended attributes for the new file were copied from the source file as follows:

Display File Attributes

```

Pathname : /pet6/aaaa
More: -
Major device . . . . . : 0
Minor device . . . . . : 0
File format . . . . . : NA
Shared AS . . . . . : 1
APF authorized . . . . . : 1
Program controlled . . . : 1
Shared library . . . . . : 0
Char Set ID/Text flag : 00000 OFF
Directory default ACL : 0
File default ACL . . . : 0
Seclabel . . . . . :

```

Using the hierarchical file system (HFS)

We provided extensive coverage of our strategy for managing the z/OS UNIX hierarchical file system (HFS), including shared HFS, in our December 2001 edition. Refer to that edition for more information.

Automount enhancement for HFS to zSeries file system (zFS) migration

We tested a new automount enhancement that eases the migration from HFS to zFS file systems. Prior to the new function, you could not use a generic automount policy to automount both HFS and zFS file systems - all the file systems had to be the same type for a given automount managed mountpoint. The enhanced HFS to zFS automount migration function allows a single automount policy to mount both HFS and zFS file systems. This will help if you want to migrate your file systems over time rather than all at once, and so have a mixture of HFS and zFS file systems in your installation.

It works like this: the automount function has changed so that when you specify either HFS or ZFS as the file system type in an automount policy, the system re-checks the data set at mount time to determine what type of data set it really is, and then directs the mount to the appropriate file system type. However, to use this function, the naming conventions of the file systems for both HFS and zFS must be the same.

The example below shows a zFS policy that we implemented to mount both HFS and zFS file systems. This policy will mount both pre-existing HFS or zFS type filesystems, but only allocates new filesystems as zFS file systems. This is the recommended policy for easing the migration to zFS file systems.

```

name *
type ZFS
filesystem OMVSSPN.<uc_name>.FS

```

```

lowercase no
allocuser space(3,2) cyl storclas(SMSOE)
mode rdwr
duration 30
delay 10

```

The next automount policy example will mount both pre-existing HFS or zFS file types as well, but will only allocate new file systems as HFS file systems:

```

name *
type HFS
filesystem OMVSPN.<uc_name>.FS
lowercase no
mode rdwr
allocuser space(3,1) cyl storclas(SMSOE)
duration 30
delay 10

```

Using the zSeries file system (zFS)

We provided extensive coverage of our strategy for setting up and managing a z/OS DFS zSeries file system (zFS) in our December 2003 edition. Refer to that edition for more information.

zFS enhancements in z/OS V1R6

The following topics describe some of the new zFS functions in z/OS V1R6 which we implemented and tested.

- “zFS parmlib search”
- “zFS performance monitoring with zfsadm (query and reset counters)” on page 166
- “HANGBREAK, zFS modify console command” on page 168

zFS parmlib search

zFS implemented a new logical parmlib search capability. We tested the following options:

Using IOEPRM00: If an IOEZPRM DD statement for specifying zFS configuration parameters is not in the started proc, the zFS will look in SYS1.PARMLIB for the existence of an IOEPRM00 member. If that member is not found, then zFS uses default settings. We created member IOEPRM00 and populated it with the settings that we use in our sysplex:

```

user_cache_size=256m
debug_setting_dsn=sys1.&SYSNAME..zfs.debug(file1)
trace_dsn=sys1.&SYSNAME..zfs.trace
trace_table_size=128m

```

Specifying IOEPRMxx: Another option is to specify one or more IOEPRMxx members of parmlib to use. The members are identified in the zFS FILESYSTYPE statement of the BPXPRMxx parmlib member. The following example shows how to specify that we want to use members IOEPRM01 and IOEPRM02 for our zFS configuration settings.

```

FILESYSTYPE TYPE(ZFS) ENTRYPPOINT(IOEFSCM) ASNAME(ZFS,'SUB=MSTR') PARM('PRM=(01,02)')

```

Using the SYSCLONE symbolic: Another option allows us to have a unique IOEPRMxx for each image by using the SYSCLONE symbolic. The following example illustrates how parmlib members would be selected if we were to start zFS on system Z0. Members IOEPRM97, IOEPRM98, IOEPRM99, and IOEPRMZ0

would be used. If a parmlib member is not found, the search for the configuration option will continue with the next parmlib member.

The maximum number of suffixes for IOEPRMxx that can be specified on the FILESYSTYPE statement is 32.

zFS performance monitoring with zfsadm (query and reset counters)

The **zfsadm query** command displays and resets zFS internal performance statistics counters and timers.

Format:

```
zfsadm query [-locking] [-reset] [-storage] [-usercache] [-iocounts]
             [-iobyaggregate] [-iobydasd] [-level] [-help]
```

Options:

- locking** Specifies that the locking statistics report should be displayed.
- reset** Specifies the report counters should be reset to zero. Should be specified with a report type.
- storage** Specifies that the storage report should be displayed.
- usercache** Specifies that the user cache report should be displayed.
- iocounts** Specifies that the I/O count report should be displayed.
- iobyaggregate** Specifies that the I/O count by aggregate report should be displayed.
- iobydasd** Specifies that the I/O count by Direct Access Storage Device (DASD) report should be displayed.
- level** Prints the level of the zfsadm command. This is useful when you are diagnosing a problem. All other valid options specified with this option are ignored.
- help** Prints the online help for this command. All other valid options specified with this option are ignored.

Note that the **-reset** option will reset the counters AFTER, not before, the display that is displayed when the option is used. For example, **zfsadm query -locking -reset** would display the locking statistics report, then reset the counters. Subsequent locking display will show statistics from counter reset.

Example: zfsadm query -locking -reset

Result:

Locking Statistics

```
Untimed sleeps:      13575   Timed Sleeps:      0   Wakeups:      13574
Total waits for locks:      22319606
Average lock wait time:      1.906 (msecs)
Total monitored sleeps:      13522
Average monitored sleep time:      5.692 (msecs)
```

Top 15 Most Highly Contended Locks

Thread Wait	Async Disp.	Spin Resol.	Pct.	Description
23683110	0	983063	99.583%	Log system map lock
3	37549	6	0.151%	Volser I/O queue lock
11852	0	20564	0.130%	Async global device lock
10321	0	192	0.42%	Vnode-cache access lock
2113	0	3705	0.23%	Transaction-cache complete list lock
1134	1116	3425	0.22%	Transaction-cache main lock
2446	0	187	0.10%	Anode bitmap allocation handle lock
1405	0	269	0.6%	Anode fileset quota lock
1437	0	4	0.5%	Async IO device lock
202	425	531	0.4%	User file cache main segment lock
829	0	81	0.3%	Anode fileset handle lock
609	0	29	0.2%	Metadata-cache buffer lock
352	0	169	0.2%	Anode file zero lock
420	0	39	0.1%	Anode file notify lock
280	0	21	0.1%	Transaction-cache active list lock
Total lock contention of all kinds: 24769331				

Top 5 Most Common Thread Sleeps

Thread Wait	Pct.	Description
13521	99.992%	Transaction allocation wait
1	0.7%	OSI cache item cleanup wait
0	0.0%	Directory Cache Buffer Wait
0	0.0%	User file cache Page Wait
0	0.0%	User file cache File Wait

Example: zfsadm query -locking

Result:

Locking Statistics

Untimed sleeps: 10 Timed Sleeps: 0 Wakeups: 10

Total waits for locks: 15898
Average lock wait time: 2.174 (msecs)

Total monitored sleeps: 10
Average monitored sleep time: 5.622 (msecs)

Top 15 Most Highly Contended Locks

Thread Wait	Async Disp.	Spin Resol.	Pct.	Description
17009	0	679	99.718%	Log system map lock
0	19	0	0.107%	Volser I/O queue lock
7	0	7	0.78%	Async global device lock
6	0	0	0.33%	Vnode-cache access lock
6	0	0	0.33%	Anode bitmap allocation handle lock
3	0	0	0.16%	Transaction-cache complete list lock
1	0	0	0.5%	Vnode lock
1	0	0	0.5%	Async IO device lock
0	0	0	0.0%	Async IO set free list lock
0	0	0	0.0%	Async IO event free list lock
0	0	0	0.0%	LVM global lock

0	0	0	0.0%	OSI Global process lock
0	0	0	0.0%	Main volume syscall lock
0	0	0	0.0%	User file cache all file lock
0	0	0	0.0%	User file cache main segment lock

Total lock contention of all kinds: 17738

Top 5 Most Common Thread Sleeps

Thread Wait	Pct.	Description
10	100.0%	Transaction allocation wait
0	0.0%	OSI cache item cleanup wait
0	0.0%	Directory Cache Buffer Wait
0	0.0%	User file cache Page Wait
0	0.0%	User file cache File Wait

Corresponding pfscctl Application Programming Interface (APIs) are also provided to retrieve these performance statistics.

- **Statistics iobyaggr Information** – The statistics iobyaggr information subcommand call contains information about the number of reads and writes and the number of bytes transferred for each aggregate.
- **Statistics iobydasd Information** – The statistics iobydasd information subcommand call contains information about the number of reads and writes and the number of bytes transferred for each DASD volume.
- **Statistics iocounts Information** – The statistics iocounts information subcommand call contains information about how often zFS performs I/O for various circumstances and how often it waits on that I/O.
- **Statistics Locking Information** – The statistics locking information subcommand call is a performance statistics operation that returns locking information.
- **Statistics Storage Information** – The statistics storage information subcommand call is a performance statistics operation that returns storage information.
- **Statistics User Cache Information** — The statistics user cache information subcommand call is a performance statistics operation that returns user cache information.

For more information on these APIs, see *z/OS Distributed File Service zSeries File System Administration*.

HANGBREAK, zFS modify console command

The following new modify console command for zFS attempts recovery for specific hang conditions: **modify procname,hangbreak**.

The **hangbreak** command causes zFS to post a failure to any requests in zFS that are waiting. This can allow the hang condition to be broken and resolved. This should only be used if you suspect that there is a hang involving zFS. The modify **zfs,query,threads** operator command is used to determine if one or more requestor threads remain in the same wait over several queries. If this command does not successfully break the hang, you need to stop or cancel zFS. If you suspect that zFS is in an infinite loop, you need to cancel zFS.

Example:

```
F ZFS,HANGBREAK
IOEZ00025I zFS kernel: MODIFY command - HANGBREAK completed successfully.
```

Issuing the su command and changing TSO identity

The following scenario describes a problem we encountered when issuing the `su` command from a z/OS UNIX System Services session, started from TSO, which did not change the TSO identity.

- We logged into TSO as a NON-superuser.
- We started an OMVS session.
- We wanted to edit a file with `oedit` that was owned by a superuser.
- We used the `su` command to switch the user identity to a superuser.
- We were not able to save the file after editing it.

After further investigation we found that if you use the OMVS interface when running a shell created by `su`, any attempt to run a TSO command results in the command running in your TSO address space. The command runs under your TSO identity, not the identity specified by `su`, unless the TSO command changes the effective UNIX identity.

`oedit` passes the effective UID of its process to the TSO session. If the EUID does not match the EUID of the TSO process, the `oedit` TSO command will attempt to set the effective UID of the TSO process to that of the shell command prior to loading the file.

One way to use `oedit` as a superuser using the `su` command is by making sure that the non-superuser id you are using is permitted to the BPX.SUPERUSER RACF facility class.

Being permitted to this facility class will not make that user a superuser but it will give that id the required permissions to become one. This can be used as a temporary solution to this problem, when one is required.

For more information, please see the z/OS Unix System Services documentation and DOC APAR OA10650.

Removing additional diagnostic data collection from OMVS CTRACE LOCK processing

We noticed increased CPU utilization and performance degradation running z/OS V1R6 with OMVS CTRACE options set at ALL or any set of options that include LOCK. This was caused by calls to query latch ownership activity when a dubbed z/OS UNIX System Services task terminates to collect additional diagnostic information. This utilization was especially noticeable when running with OMVS Heavy-weight threads, which are prevalent, for example, in Java workloads.

Due to the effect this has on the system when using the LOCK CTRACE option, the additional diagnostic data collection calls were removed from z/OS V1R6 by APAR OA10735 (PTF UA16780).

Additional collection of diagnostic latch information may be considered in future releases by other means.

Chapter 12. Using the IBM HTTP Server

This chapter describes our experiences with IBM HTTP Server V5.3.

For the most current debugging and tuning hints and tips for all supported releases of IBM HTTP Server and IBM WebSphere Application Server, see the *WebSphere Troubleshooter* (www.ibm.com/software/webservers/appserv/troubleshooter.html).

Using gskkyman support for storing a PKCS #7 file with a chain of certificates

In z/OS V1R6, the System SSL component changed the processing of the gskkyman utility to generate and manage certificates. We tested the new enhancements, which are covered in *z/OS Cryptographic Services System Secure Sockets Layer Programming*. In this section, we'll describe one small issue we encountered in using the new gskkyman support for storing certificates and their chains in a PKCS #7 file.

To test the gskkyman support for storing certificates and their chains in a PKCS #7 file, we created a certificate authority file containing the entire chain, and exported it to a PKCS #7 file with extension .p7b. So far, so good, but when we downloaded the file to our browser, we received the following message:

```
This is an invalid Security Certificate file
```

It turned out that we needed to update our HTTP server configuration file before we could download the file. We added the following AddType line:

```
AddType .p7b      application/x-x509-ca-ra-cert-chain  ebcdic 1.0
```

After adding this directive, we restarted the HTTP Server and exported the Certificate Authority to a PKCS #7 file with an extension of .p7b. Then, we were able to download the .p7b file to our browser.

Chapter 13. Using LDAP Server

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.

This chapter contains the following sections:

- “Overview of our LDAP configuration”
- “Setting up the LDAP server for RACF change logging” on page 174
- “Using the z/OS LDAP client with the Windows 2000 Active Directory service” on page 183
- “Using LDAP with Kerberos authentication” on page 184
- “LDAP Server enhancements in z/OS V1R6” on page 196

Overview of our LDAP configuration

We have a multiplatform LDAP configuration and we use both replication and referral. Figure 46 shows a high-level view of our LDAP multiplatform configuration:

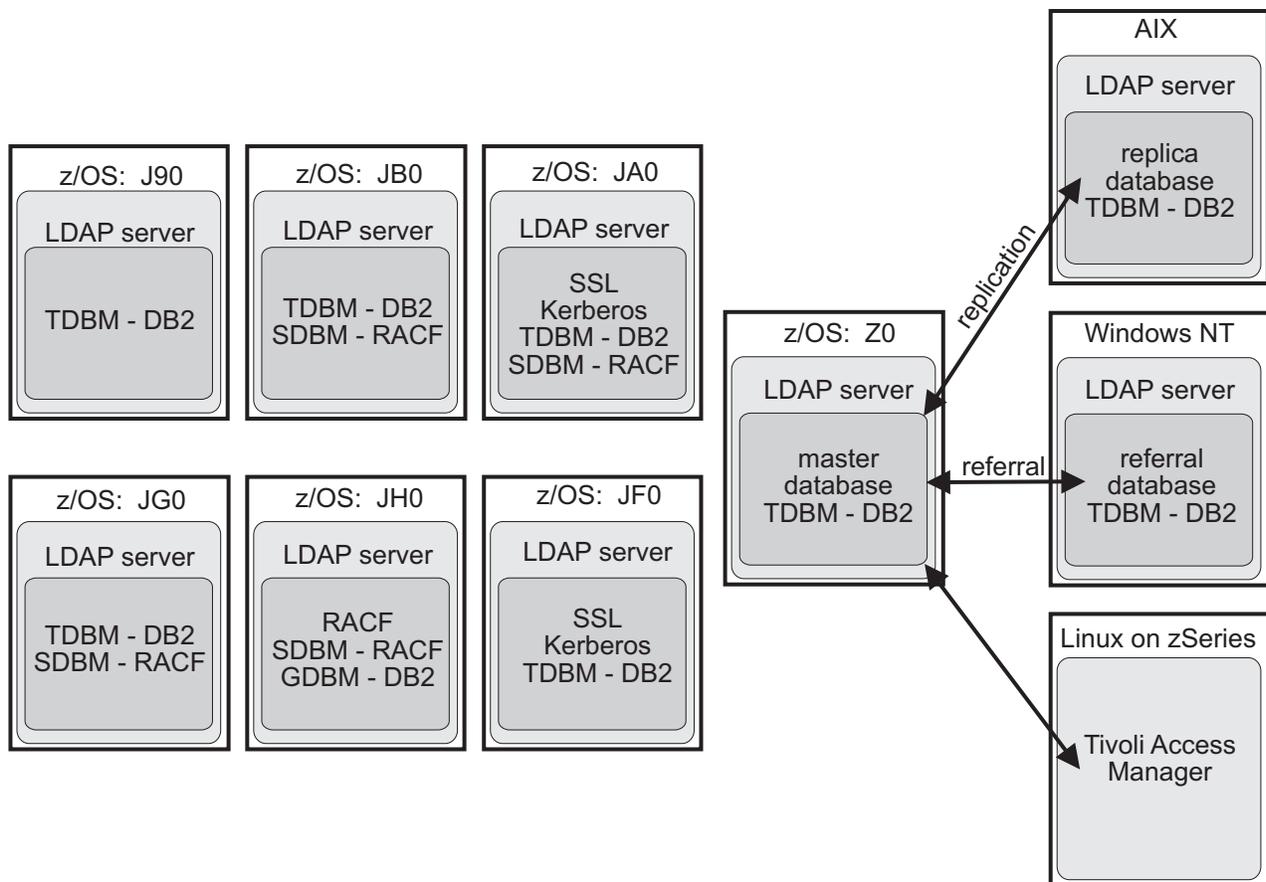


Figure 46. Overview of our LDAP configuration

Our LDAP environment includes the following features:

- **Master LDAP server:** Our master server on z/OS system Z0 has a DB2 backend database, TDBM.

LDAP Server

- **Secure LDAP servers:** We have two LDAP servers in our sysplex (on systems JA0 and JF0) that are set up for SSL secure transactions and Kerberos authentication. The servers have a TDBM backend and listen on port 636.
- **LDAP server for our RACF backend:** We have an LDAP server with an SDBM backend that connects to the RACF directory for our sysplex.
- **LDAP referral from Windows NT to z/OS:** We manage an LDAP server on Windows NT using a Web-based management tool at:
`http://NT_server_IP_address/ldap/index.html`

This LDAP server has a TDBM backend and has a general referral in its configuration file that points to our master LDAP server on z/OS. This allows a user to issue an **ldapsearch** command from the Windows NT LDAP server for an entry that is not found in that directory but that might be found in the directory on the master server on z/OS. The **ldapsearch** command returns all matching entries from both directories.

- **LDAP referral from z/OS to Windows NT:** We maintain an LDAP server referral database on Windows NT using the Directory Management Tool for Windows. This server is set up to have referral processing enabled between the master LDAP server on z/OS and the LDAP server on Windows NT.
- **LDAP replica server on AIX:** We manage an LDAP replica server on AIX using a Web-based management tool at:
`http://AIX_server_IP_address/ldap/index.html`

This LDAP server has a TDBM backend and was configured through the Web-based management tool to be a replica of the master LDAP server on z/OS.

- **Tivoli Access Manager running on Linux on zSeries:** We set up Tivoli Access Manager on a SUSE Linux image running on zSeries to enable cross-platform testing between Linux and z/OS. Tivoli Access Manager uses the z/OS LDAP Server as a backend to store user ID information that is used to authorize access for users of Tivoli Access Manager. We perform our testing by running shell scripts on Linux that create a workload to stress the master LDAP server on z/OS.

Setting up the LDAP server for RACF change logging

During our z/OS V1R5 testing, we set up and configured support for change logging in LDAP Server. This new function is delivered in Security Server LDAP APAR OA03857 and applies to z/OS V1R3 and higher.

Change logging provides the following functions:

- Provides a log of changes made to user profiles in RACF, including password changes
- Allows a client to search the log of changes
- Allows retrieval of an enveloped version of a RACF password

Log entries are stored in a new type of backend called GDBM.

Change logging also requires that the SDBM backend be configured, that LDAP Program Callable (PC) support be enabled. Support for the exploitation of this new function by RACF is provided by RACF APAR OA03853 and SAF APAR OA03854. For details and a link to the updated documentation on the Web, see LDAP APAR OA03857.

This section describes our experiences setting up and configuring change logging in our environment.

Activating change notification in RACF

We did the following to enable RACF to provide notification of changes for logging in the LDAP change log:

1. Define the RACFEVNT class profile named NOTIFY.LDAP.USER:

```
RDEFINE RACFEVNT NOTIFY.LDAP.USER
```

A generic profile can also be used.

-
2. Activated the RACFEVNT class:

```
SETRPTS CLASSACT(RACFEVNT)
```

For more information, see the documentation in RACF APAR OA03853.

Setting up the GDBM backend for the LDAP server

Note: You cannot use the LDAP configuration utility, `ldapcnf`, to configure the GDBM backend. For more information, see the documentation in LDAP APAR OA03857.

At a minimum, the GDBM backend section of the LDAP configuration file requires the following:

```
database GDBM GLDBGDBM [name]
dbuserid dbowner
servername string
```

Other options are also available to specify such things as the maximum age of a change log entry, the maximum number of entries that the change log can contain, and whether change logging is on or off (default is on).

Also, Program Callable (PC) support must be enabled in the global section of the configuration file, as follows:

```
listen ldap://:pc
```

We already had PC support enabled in our configuration file.

We did the following to set up the GDBM backend:

1. Loaded the change log schema into the LDAP server from the `ChangeLog.ldif` file:

```
ldapmodify -h ip_addr -D "cn=LDAPxxxxx" -w pw -f /path/etc/ldap/ChangeLog.ldif
```

-
2. Added the following GDBM configuration options to the `slapd.conf` configuration file. (Although SDBM was already set up on this server, we have included those options here as well).

```
# GDBM-specific CONFIGURATION SETTINGS
# -----
# -----
database gdbm GLDBGDBM
servername USIBMT6PETDB2
dbuserid GLDSRV
```

```

dsnaoini GLD.CNFOUT.JB0(DSNAOINI)
attroverflowsize 500
#####
# sdbm database definitions
#####
database sdbm GLDBSDBM
suffix "sysplex=UTCPLXJ8,o=IBM,c=US"

```

3. Started the LDAP server:

```
START LDAPxx
```

Result: We viewed the SLAPDOUT output to verify that the server startup was successful and that change logging was enabled. SLAPDOUT contained the following:

```

***** TOP OF DATA *****
GLD0022I z/OS Version 1 Release 4 Security Server LDAP Server
Starting slapd.
GLD0010I Reading configuration file /etc/ldap/slapd.conf.
GLD3135I Grant/Deny ACL support is enabled below suffixes: "CN=CHANGELOG".
GLD0244I Change logging is enabled
Logging started status (0 = off, 1 = on): 1
Limit in seconds on age of change log entries (0 = no limit): 0
Limit on the number of change log entries (0 = no limit): 0
Current number of change log entries: 0
First change number in use: 0
Last change number in use: 0
GLD0163I Backend capability listing follows:
GLD0166I Backend type: sdbm, Backend ID: SDBM BACKEND
GLD0207I SDBM BACKEND manages the following suffixes:
GLD0208I Backend suffix: SYSPLEX=UTCPLXJ8
GLD0209I End of suffixes managed by SDBM BACKEND.
GLD0165I Capability: LDAP_Backend_ID Value: SDBM BACKEND
GLD0165I Capability: LDAP_Backend_BldDateTime Value: 2003-10-21-17.46.55.000
GLD0165I Capability: LDAP_Backend_APARLevel Value: OA03857
GLD0165I Capability: LDAP_Backend_Release Value: R 4.0
GLD0165I Capability: LDAP_Backend_Version Value: V 1.0
GLD0165I Capability: LDAP_Backend_Dialect Value: DIALECT 1.0
GLD0165I Capability: LDAP_Backend_BerDecoding Value: STRING
GLD0165I Capability: LDAP_Backend_ExtGroupSearch Value: YES
GLD0165I Capability: LDAP_Backend_krbIdentityMap Value: YES
GLD0165I Capability: supportedControl Value: 2.16.840.1.113730.3.4.2
GLD0165I Capability: supportedControl Value: 1.3.18.0.2.10.2
GLD0167I End of capability listing for Backend type: sdbm, Backend ID: SDBM BACKEND
GLD0166I Backend type: gdbm, Backend ID: GDBM BACKEND
GLD0207I GDBM BACKEND manages the following suffixes:
GLD0208I Backend suffix: CN=CHANGELOG
GLD0209I End of suffixes managed by GDBM BACKEND.
GLD0165I Capability: LDAP_Backend_ID Value: GDBM BACKEND
GLD0165I Capability: LDAP_Backend_BldDateTime Value: 2003-10-21-17.47.40.000000
GLD0165I Capability: LDAP_Backend_APARLevel Value: OA03857
GLD0165I Capability: LDAP_Backend_Release Value: R 4.0
GLD0165I Capability: LDAP_Backend_Version Value: V 1.0
GLD0165I Capability: LDAP_Backend_Dialect Value: DIALECT 1.0
GLD0165I Capability: LDAP_Backend_BerDecoding Value: BINARY
GLD0165I Capability: LDAP_Backend_ExtGroupSearch Value: NO
GLD0165I Capability: LDAP_Backend_krbIdentityMap Value: NO
GLD0165I Capability: supportedControl Value: 2.16.840.1.113730.3.4.2
GLD0167I End of capability listing for Backend type: gdbm, Backend ID: GDBM BACKEND
GLD0164I Backend capability listing ended.
GLD0002I Configuration file successfully read.
GLD0189I Nonsecure communication is active for IP: INADDR_ANY, nonsecure port: 389
GLD0202I Program Call communication is active.
GLD0122I Slapd is ready for requests.
***** BOTTOM OF DATA *****

```

The LDAP server GDBM backend was successfully set up and enabled for change logging.

Testing the change logging function and the GDBM database

With change logging active, we made several changes to a RACF user ID and then tested functions to search the GDBM database, set the maximum number of change log entries, search the GDBM database anonymously, and delete change log entries.

Searching the GDBM database

Before adding any change log entries, we issued the following command to verify that the GDBM database could properly be searched:

```
ldapsearch -h ip_addr -D "racfid=XXXXX,profiletype=user,sysplex=UTCPLXJ8,o=IBM,c=US"
-w pw -b "cn=changelog" "objectclass=*
```

Result: The search displayed the following output:

```
cn=changelog
objectclass=top
objectclass=container
cn=changelog
```

Since there were no change log entries in the database yet, the search returned only the base entry of the database, as expected.

Testing the maximum number of change log entries

We did the following to test the option to limit the maximum number of change log entries:

1. Added the `changeLogMaxEntries` option to the `slapd.conf` file and specified a value of 1000, as shown:

```
# GDBM-specific CONFIGURATION SETTINGS
# -----
# -----
database gdbm GLDBGDBM
servername USIBMT6PETDB2
dbuserid GLDSRV
dsnaoini GLD.CNFOUT.JB0(DSNAOINI)
changeLogging on
changeLogMaxEntries 1000
changeLogMaxAge 86400
attroverflowsizesize 500
```

2. Made eight changes to a RACF user ID, `USER01`, which creates eight change log entries (a total of nine, including the `cn=changelog` root entry) in the GDBM backend.

3. Searched the database to verify that the change log entries were present:

```
ldapsearch -h ip_addr -D "racfid=XXXXX,profiletype=user,sysplex=UTCPLXJ8,o=IBM,c=US"
-w pw -b "cn=changelog" "objectclass=*
```

Result: The search displayed the following output:

```
cn=changelog
objectclass=top
objectclass=container
cn=changelog

CHANGENUMBER=401,CN=CHANGELOG
objectclass=CHANGELOGENTRY
```

LDAP Server

```
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=401
targetdn=RACFID=USER01,PROFILETYPE=USER,SYSPLEX=
UTCPLXJ8,0=IBM,C=US
changetime=20040127144917.638873Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETY
PE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US
```

```
CHANGENUMBER=402,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=402
targetdn=RACFID=USER01,PROFILETYPE=USER,SYSPLEX=
UTCPLXJ8,0=IBM,C=US
changetime=20040127144921.623237Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETY
PE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US
```

```
CHANGENUMBER=403,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=403
targetdn=RACFID=USER01,PROFILETYPE=USER,SYSPLEX=
UTCPLXJ8,0=IBM,C=US
changetime=20040127144925.306248Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETY
PE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US
```

```
CHANGENUMBER=404,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=404
targetdn=RACFID=USER01,PROFILETYPE=USER,SYSPLEX=
UTCPLXJ8,0=IBM,C=US
changetime=20040127144929.050827Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETY
PE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US
```

```
CHANGENUMBER=405,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=405
targetdn=RACFID=USER01,PROFILETYPE=USER,SYSPLEX=
UTCPLXJ8,0=IBM,C=US
changetime=20040127144933.085019Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETY
PE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US
```

```
CHANGENUMBER=406,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
```

```

changenumber=406
targetdn=RACFID=USER01,PROFILETYPE=USER,SYSPLEX=
UTCPLXJ8,0=IBM,C=US
changetime=20040127144938.965894Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETY
PE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US

```

```

CHANGENUMBER=407,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=407
targetdn=RACFID=USER01,PROFILETYPE=USER,SYSPLEX=
UTCPLXJ8,0=IBM,C=US
changetime=20040127144942.378976Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETY
PE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US

```

```

CHANGENUMBER=408,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=408
targetdn=RACFID=USER01,PROFILETYPE=USER,SYSPLEX=
UTCPLXJ8,0=IBM,C=US
changetime=20040127144945.559614Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETY
PE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US

```

-
4. Changed the value of the `changeLogMaxEntries` option in the `slapd.conf` file from 1000 to 5, as shown:

```

# GDBM-specific CONFIGURATION SETTINGS
# -----
# -----
database gdbm GLDBGDBM
servername USIBMT6PETDB2
dbuserid GLDSRV
dsnaoini GLD.CNFOUT.JB0(DSNAOINI)
changeLogging on
changeLogMaxEntries 5
changeLogMaxAge 86400
attroverflowsizesize 500

```

-
5. Recycled the LDAP server.

-
6. Issued the LDAP search again:

```

ldapsearch -h ip_addr -D "racfid=XXXXX,profiletype=user,sysplex=UTCPLXJ8,o=IBM,c=US"
-w pw -b "cn=changelog" "objectclass=*"

```

Result: The search displayed the following output:

```

cn=changelog
objectclass=top
objectclass=container
cn=changelog

```

```

CHANGENUMBER=405,CN=CHANGELOG

```

```
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=405
targetdn=RACFID=USER01,PROFILETYPE=USER,SYSPLEX=
UTCPLXJ8,0=IBM,C=US
changetime=20040127144933.085019Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETY
PE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US
```

```
CHANGENUMBER=406,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=406
targetdn=RACFID=USER01,PROFILETYPE=USER,SYSPLEX=
UTCPLXJ8,0=IBM,C=US
changetime=20040127144938.965894Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETY
PE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US
```

```
CHANGENUMBER=407,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=407
targetdn=RACFID=USER01,PROFILETYPE=USER,SYSPLEX=
UTCPLXJ8,0=IBM,C=US
changetime=20040127144942.378976Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETY
PE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US
```

```
CHANGENUMBER=408,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=408
targetdn=RACFID=USER01,PROFILETYPE=USER,SYSPLEX=
UTCPLXJ8,0=IBM,C=US
changetime=20040127144945.559614Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETY
PE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US
```

Note that change log entries 401 through 404 have been deleted. There were now a total of five entries (including the cn=changelog root entry) in the database, as specified by the changeLogMaxEntries value.

The option to limit the maximum number of change log entries worked successfully.

Searching the GDBM database anonymously

We did the following to enable and test the ability to anonymously search the GDBM backend:

1. Performed an anonymous search to make sure this feature was not already enabled:

```
ldapsearch -h ip_addr -b "cn=changelog" "objectclass=*"
```

Result: The search displayed no output and simply returned to the command line.

2. Performed an administrative search to make sure the database could properly be searched:

```
ldapsearch -h ip_addr -D "racfid=XXXXX,profiletype=user,sysplex=UTCPLXJ8,o=IBM,c=US"
-w pw -b "cn=changelog" "objectclass=*"

```

Result: The search displayed the following output:

```
cn=changelog
objectclass=top
objectclass=container
cn=changelog

```

```
CHANGENUMBER=501,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=501
targetdn=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C=US
changetime=20040203154225.041549Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US

```

```
CHANGENUMBER=502,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=502
targetdn=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C=US
changetime=20040203154229.595015Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US

```

```
CHANGENUMBER=503,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=503
targetdn=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C=US
changetime=20040203154237.589303Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US

```

```
CHANGENUMBER=504,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=504
targetdn=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C=US
changetime=20040203154242.123712Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C
=US

```

3. Created a file, cl.ldif, with the following contents to modify the ACL to allow anonymous searches:

```
1 dn: cn=changelog
2 changetype: modify
3 replace: aclentry
4 aclentry:cn=Anybody:normal:rsc:sensitive:rsc:critical:rsc:system:rsc
```

4. Loaded the new information from the cl.ldif file into the database:

```
ldapmodify -h ip_addr -D "racfid=XXXXX,profiletype=user,sysplex=UTCPLXJ8,o=IBM,c=US"
-w pw -f cl.ldif
```

Result: The changes were successfully loaded.

5. Performed an anonymous search again:

```
ldapsearch -h ip_addr -b "cn=changelog" "objectclass=*"
```

Result: This time, the search displayed the database contents:

```
cn=changelog
objectclass=top
objectclass=container
cn=changelog
```

```
CHANGENUMBER=501,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=501
targetdn=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,O=IBM,C=US
changetime=20040203154225.041549Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,O=IBM,C=US
```

```
CHANGENUMBER=502,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=502
targetdn=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,O=IBM,C=US
changetime=20040203154229.595015Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,O=IBM,C=US
```

```
CHANGENUMBER=503,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=503
targetdn=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,O=IBM,C=US
changetime=20040203154237.589303Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,O=IBM,C=US
```

```
CHANGENUMBER=504,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=504
targetdn=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,O=IBM,C=US
changetime=20040203154242.123712Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,O=IBM,C=US
```

Anonymous searches of the database worked successfully.

Deleting change log entries

We did the following to test the ability to delete unwanted change log entries from the database:

1. Performed a search for a specific change log entry:

```
ldapsearch -h ip_addr -b "changenumber=604, cn=changelog" "objectclass=*
```

Result: The search displayed the following:

```
CHANGENUMBER=604,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=604
targetdn=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C=US
changetime=20040209151933.319305Z
changetype=MODIFY
ibm-changeinitiatorsname=RACFID=XXXXX,PROFILETYPE=USER,SYSPLEX=UTCPLXJ8,0=IBM,C=US
```

2. Issued the following command to delete the specific change log entry:

```
ldapdelete -h ip_addr -D "racfid=XXXXX,profiletype=user,sysplex=UTCPLXJ8,o=IBM,c=US"
-w pw "changenumber=604, cn=changelog"
```

Result: The command completed successfully.

3. Searched again for the specific change log entry to verify that it was gone:

```
ldapsearch -h ip_addr -b "changenumber=604, cn=changelog" "objectclass=*
```

Result: The search displayed the following:

```
ldap_search: No such object
ldap_search: matched: cn=changelog
ldap_search: additional info: R004026 Entry changenumber=604, cn=changelog not
found in the database. (tdbm_search.c|1.74.3.2|841)
```

The change log entry was successfully deleted from the database.

Using the z/OS LDAP client with the Windows 2000 Active Directory service

We had a request from our colleagues in z/OS LDAP development to test the z/OS LDAP client with the Microsoft Windows 2000 Active Directory service using a Kerberos authentication bind. This was to help validate a fix they were working on involving the use of Kerberos authentication to bind to IBM Directory Server. In order to do this, we first wanted to ensure that we could perform a search using a simple LDAP bind between the z/OS LDAP client and Active Directory.

We did not need to do anything special to set up or enable this process. We already had access to a Windows 2000 server that had Active Directory enabled and had data loaded. The hardest part was determining the format of the Active Directory distinguished name. Fortunately, we were able to get some advice from a colleague who has some experience with Active Directory. We were eventually able to successfully perform a search from z/OS against the Active Directory, as shown in the following example:

LDAP Server

Example: We issued the following **ldapsearch** command (as a single line) from the z/OS UNIX shell command line against the Windows 2000 Active Directory:

```
ldapsearch -h win2k_ip_addr
-D "CN=Sue Marcotte,CN=Users,DC=kerberos,DC=xxx,DC=yyy,DC=ibm,DC=com" -w password
-b "CN=Sue Marcotte,CN=Users,DC=kerberos,DC=xxx,DC=yyy,DC=ibm,DC=com" "objectclass=*" name
```

Result: As expected, we received the following response:

```
CN=Sue Marcotte,CN=Users,DC=kerberos,DC=xxx,DC=yyy,DC=ibm,DC=com
name=Sue Marcotte
```

Using LDAP with Kerberos authentication

We have implemented a new LDAP workload that exploits binding with Kerberos authentication. (Our December 2002 edition describes how we enabled the z/OS LDAP Server for Kerberos authentication.) This workload has uncovered a couple of problems that we would like to note.

We used the following documentation to help us investigate and diagnose these problems:

- *z/OS Integrated Security Services LDAP Client Programming, SC24-5924*
- *z/OS Integrated Security Services LDAP Server Administration and Use, SC24-5923*
- *z/OS Integrated Security Services Network Authentication Service Administration, SC24-5926*

Problems we experienced with our workload

The following are the problems that we experienced with our LDAP workload using Kerberos authentication:

Abend 0C6 in LDAP Server

We experienced an abend 0C6 in LDAP Server under the following conditions:

- LDAP Server successfully started but failed to configure a TDBM backend. (There is an SDBM backend.)
- An **ldapsearch** command to the TDBM backend is issued to the server using a Kerberos authentication bind.

This caused LDAP Server to generate a CEEDUMP and fail with an abend 0C6. Below is a portion of the traceback that helped to identify the error.

```
Traceback:
 DSA Addr Program Unit PU Addr PU Offset Entry E Addr E Offset Statement Load Mod Service Status
26EE74B8 CEEHDSP 26955430 +00003F42 CEEHDSP 26955430 +00003F42 CEEPLPKA D1515 Call
26EE69C8 27CF9D68 +2870A3DB addAltDN(_Connection*,_strbuf*,_Backend*)
26EE68A8 27CF9378 +000002D0 storeBindInfo(_Connection*,_Operation*)
26EE6788 27CF8548 +00000612 krbBind(_Connection*,_Operation*,_berval*)
26EE66A0 2686AB68 +0000015A SaslBindGssapi::doBindPart1(_strbuf*)
26EE6550 2680A430 +000010A2 do_bind(_Connection*,_Operation*)
26EE63D8 26838550 +00000F38 process_request
26EE62A0 2683F908 +0000086E caMReceiveCB(void*,void*,int,int)
26EE61A8 26849E00 +0000020E caSslAsyncBerGetNextCB(sockbuf*)
26EE60E8 05F50398 +00000164 asyncBerGetNext
26EE6008 05F513E0 +00000086 asyncBerGetNextCB(sockbuf*)
26EE5F30 05F50D48 +00000136 async_get_tag(sockbuf*,unsigned long*,void*)(sockbuf*)
```

26EE5E68	05F51308	+0000005E	async_get_tagCB(sockbuf*)	05F50D48	+00000136		GLDNMC03	Call
26EE5D80	05F50A38	+0000026C	async_BerRead(sockbuf*,char*,long,long*,void*)(sockbuf*)	05F51308	+0000005E		GLDNMC03	Call
26EE5CB8	05F51220	+00000072	async_BerRead_filbufCB(sockbuf*)	05F50A38	+0000026C		GLDNMC03	Call
26EE5BF0	05F51118	+0000008A	async_ber_filbufCB(sockbuf*,int)	05F51220	+00000072		GLDNMC03	Call
26EE5AF0	2684A6A0	+00000174	caSslLowerReceiveCB(void*,void*,int,int)	05F51118	+0000008A		GLDNMC03	Call
26EE59E8	26856468	+000003D8	caInetReceiveCB	2684A6A0	+00000174		GLDNM005	Call
26EE5930	26AC8CD8	+0000001A	@GETFN	26856468	+000003D8		GLDNM005	Call
26EE57F0	05FEE3F0	+000009A4	async_service_thread	26AC8C30	+000000C2		CEEEV003	Call
26EE5718	05FE34A8	+0000022E	osi_thread_first	05FEE3F0	+000009A4		GLDNM035	Call
7F599E78	CEEOPCMM	0000E4D0	CEEOPCMM	05FE34A8	+0000022E		GLDNM035	Call
				0000E4D0	+00000914		CEEBINIT	D1515

Condition Information for Active Routines

Condition Information for (DSA address 26EE69C8)

CIB Address: 26EE7DF8

Current Condition:

CEE0198S The termination of a thread was signaled due to an unhandled condition.

Original Condition:

CEE3206S The system detected a specification exception (System Completion Code=0C6).

Location:

Program Unit: Entry: addAltDN(_Connection*,_strbuf*,_Backend*)

Statement: Offset: +2870A3DB

APAR OA07015 addresses this problem.

Abend 0C4 in gss_release_buffer in z/OS LDAP client

We experienced an intermittent problem in which the z/OS LDAP client, while binding to the z/OS LDAP server using Kerberos authentication, generated a CEEDUMP with an exception in gss_release_buffer and failed with an abend 0C4. The CEEDUMP file appeared in the HFS under the directory from which the client was running.

Below is a portion of the traceback that helped to identify the error.

Traceback:	DSA Addr	Program Unit	PU Addr	PU Offset	Entry	E Addr	E Offset	Statement	Load Mod	Service	Status
	29D63140		29E41900	+00000234	eim_snap_dump	29E41900	+00000234	77 *PATHNAM		HIT7708	Call
	29D63028		29E41C28	+000006A6	eim_exc_handler	29E41C28	+000006A6	294 *PATHNAM		HIT7708	Call
	29D62F70		29A51838	+0000001A	@GETFN	29A51790	+000000C2	CEEEV003			Call
	29D5FE30	CEEHDSP	298EBA50	+000024D8	CEEHDSP	298EBA50	+000024D8	CEEPLPKA		D1515	Call
	29D5F3C0		078001C0	+0000016A	gss_release_buffer	078001C0	+0000016A				Exception
	29D5F2E8		08F96798	+000000B6	ldap_gss_release_buffer(gss_buffer_desc_struct*)	08F96798	+000000B6			GLDNMC03	Call
	29D5F1A0		08EF0858	+00000550	ldap_krb5_authenticate(ldap*,berval*,_LDAPControl**,_LDAPCon	08EF0858	+00000550			GLDNMC01	Call
	29D5EE68		08EEFB50	+00000AA6	ldap_sasl_bind_krb5_s_direct	08EEFB50	+00000AA6			GLDNMC01	Call
	29D5EDA0		08EC6AB8	+00000118	ldap_sasl_bind_s_direct	08EC6AB8	+00000118			GLDNMC01	Call
	29D5ECA8		08EC58E0	+00000326	ldap_sasl_bind_s	08EC58E0	+00000326			GLDNMC01	Call
	29D5EBA0		29E03EF8	+000000E2	eimHandleInt::kerberosBind(ldap*,unsigned long*,eimErr*)	29E03EF8	+000000E2	2936 *PATHNAM		HIT7708	Call
	29D5E9F8		29E01498	+000005A0	eimHandleInt::connect2(eimLdapInfo*,EimConnectInfo*,eimClean	29E01498	+000005A0	2435 *PATHNAM		HIT7708	Call
	29D5E8A8		29E03248	+000002A8	eimHandleInt::setMaster2(char*,int,eimErr*)	29E03248	+000002A8	2072 *PATHNAM		HIT7708	Call
	29D5E7C8		29E03130	+0000008C	eimHandleInt::connectToMaster(EimConnectInfo*,eimErr*)	29E03130	+0000008C	1433 *PATHNAM		HIT7708	Call
	29D5E6F8		29E39430	+00000082	qsy_eimConnectToMaster(eimHandleInt*,EimConnectInfo*,eimErr*	29E39430	+00000082	714 *PATHNAM		HIT7708	Call
	29D5E530		29E281C8	+00000410	eimConnectToMaster	29E281C8	+00000410	3050 *PATHNAM		HIT7708	Call
	29D5E368		29809840	+00000D28	connectEIM	29809840	+00000D28	924 *PATHNAM		HIT7708	Call

LDAP Server

```
29D5E210          2980D1C0 +000002FC main          2980D1C0 +000002FC          184 *PATHNAM HIT7708 Ca11
29D5E0F8          29C2A146 +000000B4 EDCZMINV        29C2A146 +000000B4          CEEEV003          Ca11
29D5E030 CEEBBEXT      298BBAA0 +000001A6 CEEBBEXT        298BBAA0 +000001A6          CEEPLPKA D1515    Ca11
```

Condition Information for Active Routines

Condition Information for (DSA address 29D5F3C0)

CIB Address: 29D60770

Current Condition:

CEE3204S The system detected a protection exception (System Completion Code=0C4).

Original Condition:

CEE3204S The system detected a protection exception (System Completion Code=0C4).

Location:

Program Unit: Entry: gss_release_buffer

Statement: Offset: +0000016A

Machine State:

ILC..... 0004 Interruption Code..... 0004

PSW..... 078D1400 8780032E

GPR0..... 29D99CB0 GPR1..... 29D5F458 GPR2..... 29D99CB0 GPR3..... 078001FA

GPR4..... 0000E748 GPR5..... 00000000 GPR6..... 29D99D00 GPR7..... 00000000

GPR8..... 29D5F3BC GPR9..... 29D8E010 GPR10.... 29E6B130 GPR11.... 29D8A840

GPR12.... 2983B7A8 GPR13.... 29D5F3C0 GPR14.... 8780032A GPR15.... 0003032A

APAR OA07090 addresses this problem.

Setting up SSL client and server authentication between z/OS LDAP V1R6 server/client and Sun ONE Directory Server 5.2 server/client

We set up SSL Client and Server authentication between the Sun ONE Directory Server 5.2 and z/OS LDAP Server/Client V1R6, when we were evaluating the Sun ONE Directory Server 5.2.

This topic describes how to setup:

- SSL Server Authentication between a Sun ONE Directory Server 5.2 and a z/OS LDAP V1R6 Client
- SSL Server Authentication between a z/OS V1R6 LDAP Server and a Sun ONE Directory Server 5.2 Client
- SSL Server and Client authentication between a Sun ONE Directory Server 5.2 and a z/OS LDAP V1R6 Client
- SSL Server and Client authentication between a z/OS V1R6 LDAP Server and a Sun ONE Directory Server 5.2 Client

Assumptions:

- You have a z/OS LDAP v1r6 LDAP Server setup. Populated with entries and accepting NON-secure communications.
- You have a Sun ONE Directory Server 5.2 setup on a Sun Solaris 9.0 operating system. Populated with entries and accepting NON-secure communications.

Since we are in a test environment, we chose to use z/OS's gskkyman utility to setup our own Certificate Authority (CA).

In order to act as a CA, a certificate key database and a CA certificate were created following these instructions:

- Create a CA Certificate Key Database and a CA certificate:
 - Start z/OS's gskkyman utility from an OMVS shell (gskkyman)
 - Follow the instructions on the screen to create a new key database. For example, myCA.kdb.
 - Once the certificate key database is created you will be at the "Key Management Menu"
 - Pick option 6, "Create a Self-signed Certificate"

- | – For this example we picked option 1, “CA Certificate with 1024-bit RSA Key”,
- | you may choose to pick any of the CA certificates listed.
- | – We filled out the rest of the information requested as follows:
- | – Enter label (press ENTER to return to menu): myCAcert
- | – Enter subject name for certificate
- | – Common name (required): My CA
- | – Organizational unit (optional): myUnit
- | – Organization (required): myOrg
- | – City/Locality (optional): Pok
- | – State/Province (optional): NY
- | – Country/Region (2 characters - required): US
- | – Enter number of days certificate will be valid (default 365): <enter>
- | – Enter 1 to specify subject alternate names or 0 to continue: 0
- | – You should see:
- | – Certificate created.
- | – Press ENTER to continue.
- | – Once you hit enter you will be back at the “Key Management Menu”. Pick
- | option 1, “Manage Keys and Certificates”
- | – Pick myCAcert from the list of the certificates, and then pick option 3, “Set Key
- | as Default”. Next, pick option 6, “Export Certificate To a File”
- | – We picked “Base64 ASN.1 DER” as the Export File Format for this example and
- | hit enter
- | – Export File Name: myCAcert

| Now we have a CA certificate and a CA certificate key database. We will be using

| them for signing certificate requests from our servers and clients, therefore act as

| our own CA.

| In the following examples, whenever a CA certificate is needed we will use

| myCAcert and whenever the CA certificate key database is needed we will use

| myCA.kdb. In a production environment, unless you are acting as your own CA, you

| would want to send the certificate requests that you create for your servers and

| clients to the CA in order to get them signed.

| Also, remember that the following are examples only. Not all possible scenarios are

| considered, only basic setups are explained. Make sure to review the Sun ONE

| Directory Server 5.2 and z/OS LDAP V1R6 documentation for details, especially

| when setting up in a production environment.

- | 1. SSL Server Authentication between a Sun ONE Directory Server 5.2 and a z/OS
- | LDAP V1R6 Client:
 - | a. Enable Sun ONE Directory Server 5.2 for SSL Communications:
 - | • Transfer myCAcert over to the Solaris system hosting the Sun ONE
 - | Directory Server 5.2 and install it, using the Sun ONE Directory Server
 - | 5.2 Administration document.
 - | • Using the Sun ONE Directory Server 5.2 Administration document, create
 - | a certificate request for the directory server. The Common Name field in
 - | the request should be the name of your server (for example: the IP
 - | address).

LDAP Server

- Get the certificate request signed by your CA. For this example, since we are acting as our own CA, transfer the request to the z/OS system where myCA.kdb is located.
 - Sign the certificate request using gskkyman. For example:

```
gskkyman -g -x 365 -cr sunServer.req -ct sunServerCert -k /etc/ldap/myCA.kdb
```
 - sunServerCert is now created. Transfer it back to the Sun system, and install the server certificate.
- b. Create a certificate key database for the z/OS LDAP client and install the CA certificate myCAcert in that database:
- Start gskkyman on z/OS
 - Follow the instructions to create a new key database
For this example: z0Sclient.kdb
 - Pick option 7, “*Import a Certificate*”
 - Enter the certificate file name: myCAcert
 - Label it as you wish, we labeled it as myCAcert
 - Hit enter and the certificate is imported
 - From the “*Key Management Menu*” pick option 2, “*Manage Certificates*” then pick myCAcert. Next pick option 2 “*Set Certificate Trust Status*” to make sure that this CA certificate is trusted.
- c. Next, we tested our setup. We did this by running an ldapsearch command, from z/OS against the Sun ONE Directory Server, with the following options:

```
ldapsearch -h <host name> -p <secure port> -Z -K <client kdb> -P <client key database  
password> -b <search string> <filter>
```

For example:

```
ldapsearch -h solarisBox -p 636 -Z -K /u/test/keys/z0Sclient.kdb -P secret  
-b "cn=John Doe, o=Your Company, c=US" objectclass=*
```

Note: You might not receive the search results you are expecting or any error messages. That means that the bind was successful but you don't have access to see the information you requested. This depends on how your server is setup. On the other hand binding with the administrator DN should give you results.

```
ldapsearch -h solarisBox -p 636 -D "cn=LDAP Administrator" -w secret -Z  
-K /u/test/keys/z0Sclient.kdb -P secret -b "cn=John Doe, o=Your Company, c=US"  
objectclass=*
```

2. SSL Server Authentication between a z/OS V1R6 LDAP Server and a Sun ONE Directory Server 5.2 Client

Note: Sun ONE Directory Server 5.2 provides you with a tool called certutil. See Server Administration document for more information on this tool.

- a. Create a certificate key database for Sun's LDAP client, using certutil. For example:

```
certutil -N -d /export/home/test/keys/ -P client-
```

Look up how to use certutil for detailed instructions and all the available options.

This command will create two key databases:

```
client-cert7.db  
client-key3.db
```

- b. Transfer the CA certificate myCAcert, which we created, to the Solaris system. Install the CA certificate to the client key database, for example:

```
certutil -A -n myCAcert -t "TC,," -a -d /export/home/test/keys -P client- -I  
/export/home/test/keys/myCAcert
```

- c. Create a certificate key database, import the CA certificate into it and create a new certificate request for the z/OS LDAP server using gskkyman:

- Start gskkyman.
- Create a new key database using the instructions. For example:
z0Sserver.kdb
- Pick option 7, “*Import a Certificate*”.
- Import myCAcert (the CA certificate we created earlier).
- Pick option 4, “*Create New Certificate Request*”
- Follow the instructions to create a new certificate request for your z/OS LDAP server. For this example I used the following values:

```
Enter certificate type (press ENTER to return to menu): 1
Enter request file name (press ENTER to return to menu): z0Sserver.req
Enter label (press ENTER to return to menu): z0SserverCert
Enter subject name for certificate
Common name (required): cn=<server's IP goes here>
Organizational unit (optional): my0u
Organization (required): my0rg
City/Locality (optional): Pok
State/Province (optional): NY
Country/Region (2 characters - required): US
```

Note:

The “Common name” value above must be in that format: cn=server name or IP.

- d. Exit out of gskkyman. Now you must sign the certificate request using gskkyman and the CA certificate we created earlier:

```
gskkyman -g -x 365 -cr z0Sserver.req -ct z0SserverCert -k /u/test/keys/myCA.kdb
```

Now you have a certificate for your server named z0SserverCert.

- e. Install the certificate into your server key database. Start gskkyman, open z0Sserver.kdb, pick option 5, “*Receive requested certificate or a renewal certificate*”, enter filename when requested to, z0SserverCert and hit enter.
- f. Enable z/OS V1R6 LDAP Server for SSL communications following the instructions in “Integrated Security Services LDAP Server Administration and Use” document, use z0Sserver.kdb as your server’s certificate key database.
- g. Next, we tested our setup. We did this by running an ldapsearch command, from the Sun Solaris system against the z/OS LDAP server, with the following options:

```
ldapsearch -h <host> -p <secure port> -P <client's cert7 db>
-b <search string> <filter>
```

For example:

```
ldapsearch -h z0Saddress -p 636 -P /export/home/test/keys/client-cert7.db
-b "cn=John Doe,o=Your Company,c=US" objectclass=*
```

3. SSL Server and Client authentication between a Sun ONE Directory Server 5.2 and a z/OS LDAP V1R6 Client

- a. Make sure you have SSL Server Authentication setup between the Sun ONE Directory Server and z/OS LDAP Client. If not, follow the instructions in 1 on page 187.
- b. Now you should have:

```
Client key database on z/OS system: z0Sclient.kdb
Server key database on Sun Solaris system: <server-name>-cert7.db and
<server-name>-key3.db
Both databases should contain a copy of: myCAcert CA certificate
```

LDAP Server

- c. Next, we created a client certificate (for z/OS LDAP Client) request and signed it by way of the CA certificate we created earlier. In this example we created a client certificate for “cn=John Doe,o=Your Company,c=US” entry in the Sun ONE directory server.

- Start gskkyman
- Open z0ScIient.kdb
- Pick option 4, “*Create New Certificate Request*”
- Follow the instructions to create a new certificate request for “cn=John Doe,o=Your Company,c=US”. For this example we used the following values:

```
Enter certificate type (press ENTER to return to menu): 1
Enter request file name (press ENTER to return to menu): JohnDoe.req
Enter label (press ENTER to return to menu): JohnDoeCert
Enter subject name for certificate
Common name (required): John Doe
Organizational unit (optional):
  Organization (required): Your Company
  City/Locality (optional):
  State/Province (optional):
  Country/Region (2 characters - required): US
```

Once the certificate request is created, sign the request:

```
gskkyman -g -x 365 -cr JohnDoe.req -ct JohnDoeCert -k /u/test/keys/myCA.kdb
```

Now you have a certificate named JohnDoeCert.

- d. Import the client certificate JohnDoeCert into your client key database z0ScIient.kdb:

- Start gskkyman
- Open z0ScIient.kdb using the instructions:
- Pick option 7, “*Import a Certificate*”
- Import JohnDoeCert

- e. Next, we setup the Sun ONE Directory Server mappings so that John Doe can bind to the server using his client certificate, from z/OS.

- Locate certmap.conf on your Sun Solaris system
- We didn’t edit the default mapping scheme but created a new one; something like this:

```
certmap myCA          cn=My CA, ou=MyUnit, o=MyOrg, l=Pok, st=NY, c=US
myCA:DNComps         o, c
myCA:FilterComps     cn
myCA:verifycert      off
```

For more information on mappings see Sun ONE Directory Server 5.2 documentation.

- f. Make changes to the access control to test our setup:

Please note that the actions below were taken just to test this setup easily. To better understand how to manage access controls make sure to see: “*Sun ONE Directory Server 5.2 Administration Guide: Ch 6- Managing Access Control*”.

“Depending on the ACIs defined for the directory, for certain operations, you need to *bind* to the directory. *Binding* means logging in or authenticating yourself to the directory by providing a bind DN and password, or, if using SSL, a certificate. The credentials provided in the bind operation, and the circumstances of the bind determine whether access to the directory is allowed or denied.”

From the console, you can set this permission by doing the following:

- 1) On the Directory tab, right click the o=Your Company,c=US node in the left navigation tree, and choose Set Access Permissions from the pop-up menu to display the Access Control Manager.
- 2) Click New to display the Access Control Editor.
- 3) On the Users/Groups tab, in the ACI name field, type "John Doe Bound - Read, Search, Compare". Click add and find John Doe to add it to the list of users granted access permission.
- 4) On the Rights tab, tick the checkboxes for read, compare, and search rights. Make sure the other checkboxes are clear.
- 5) On the Targets tab, click This Entry to display the o=Your Company,c=US suffix in the target directory entry field. In the attribute table, locate the userPassword attribute and clear the corresponding checkbox. All other checkboxes should be ticked.

Note: This task is made easier if you click the Name header to organize the list of attributes alphabetically.

- 6) Click OK in the Access Control Editor window.

g. Next, we tested our setup:

```
ldapsearch -h sunServer -p 636 -Z -S EXTERNAL -N JohnDoeCert -K
/u/test/keys/client.kdb -P secret -b "cn=John Doe,o=Your Company,c=US" objectclass=*
```

4. SSL Server and Client authentication between a z/OS V1R6 LDAP Server and a Sun ONE Directory Server 5.2 Client

a. Make sure you have SSL Server Authentication setup between the z/OS LDAP Server and Sun LDAP Client. If not, follow the instructions in 2 on page 188.

b. Now you should have:

```
Client key databases on Sun Solaris system: client-cert7.db and client-key3.db
Server key database on z/OS system: z0Sserver.kdb
Both databases should contain a copy of: myCAcert CA certificate
```

c. Next, we created a client certificate request (for Sun LDAP Client) and signed it by way of the CA certificate we created earlier:

This example created a client certificate request called JohnDoe.req for the entry "cn=John Doe,o=Your Company,c=US" in directory /export/home/test/keys, and the request is associated with client- set of certificate key databases.

```
certutil -R -s "cn=John Doe,o=Your Company,c=US" -a -o JohnDoe.req
-d /export/home/test/keys -P client-
```

Once the certificate request is created, transfer the request to z/OS. Sign the request:

```
gskkyman -g -x 365 -cr JohnDoe.req -ct JohnDoeCert -k /u/test/keys/myCA.kdb
```

We now have a certificate named JohnDoeCert, and transferred it back to our Sun Solaris system.

d. Install the client certificate JohnDoeCert into your client key database client-cert7.db:

```
certutil -A -n "JohnDoeCert" -t "u,," -a -i /export/home/test/keys/JohnDoeCert -d
/export/home/test/keys/ -P client-
```

e. Next, we tested our setup.

Caution: Do not use the -D and -w options with client authentication, as the bind operation will use the authentication credentials specified with --D and -w instead of the certificate credentials desired. For example;

```
ldapsearch -h x.xx.xx.xxx -p 636 -Z -P /export/home/test/keys/client-cert7.db -N  
JohnDoeCert -K /export/home/test/keys/client-key3.db -W secret -b "cn=John Doe,  
o=Your Company,c=US" "objectclass=*
```

Setting up SSL client and server authentication between z/OS LDAP V1R6 server/client and IBM Tivoli Directory Server 5.2 server/client

This topic describes how to setup:

- SSL Server Authentication between an IBM Tivoli Directory Server 5.2 and a z/OS LDAP V1R6 Client
- SSL Server Authentication between a z/OS V1R6 LDAP Server and an IBM Tivoli Directory Server 5.2 Client
- SSL Server and Client authentication between an IBM Tivoli Directory Server 5.2 and a z/OS LDAP V1R6 Client
- SSL Server and Client authentication between a z/OS V1R6 LDAP Server and an IBM Tivoli Directory Server 5.2 Client

Assumptions:

- You have a z/OS LDAP V1R6 LDAP Server setup. Populated with entries and accepting NON-secure communications.
- You have an IBM Tivoli Directory Server 5.2 setup on a SUSE Sles 8, for Linux on zSeries, operating system. Populated with entries and accepting NON-secure communications.

Since we are in a test environment, we chose to use z/OS's gskkyman utility to setup our own Certificate Authority (CA).

In order to act as a CA, a certificate key database and a CA certificate were created following these instructions:

- Create a CA Certificate Key Database and a CA certificate:
 - Start z/OS's gskkyman utility from an OMVS shell (gskkyman)
 - Follow the instructions on the screen to create a new key database. For example, myCA.kdb.
 - Once the certificate key database is created you will be at the "Key Management Menu"
 - Pick option 6, "Create a Self-signed Certificate"
 - For this example we picked option 1, "CA Certificate with 1024-bit RSA Key", you may choose to pick any of the CA certificates listed.
 - We filled out the rest of the information requested as follows:
 - Enter label (press ENTER to return to menu): myCAcert
 - Enter subject name for certificate
 - Common name (required): My CA
 - Organizational unit (optional): myUnit
 - Organization (required): myOrg
 - City/Locality (optional): Pok
 - State/Province (optional): NY
 - Country/Region (2 characters - required): US
 - Enter number of days certificate will be valid (default 365): <enter>
 - Enter 1 to specify subject alternate names or 0 to continue: 0
 - You should see:

- Certificate created.
- Press ENTER to continue.
- Once you hit enter you will be back at the “*Key Management Menu*”. Pick option 1, “*Manage Keys and Certificates*”
- Pick myCAcert from the list of the certificates, and then pick option 3, “*Set Key as Default*”. Next, pick option 6, “*Export Certificate To a File*”
- We picked “Base64 ASN.1 DER” as the Export File Format for this example and hit enter
- Export File Name: myCAcert

Now we have a CA certificate and a CA certificate key database. We will be using them for signing certificate requests from our servers and clients, therefore act as our own CA.

In the following examples, whenever a CA certificate is needed we will use myCAcert and whenever the CA certificate key database is needed we will use myCA.kdb. In a production environment, unless you are acting as your own CA, you would want to send the certificate requests that you create for your servers and clients to the CA in order to get them signed.

Also, remember that the following are examples only. Not all possible scenarios are considered, only basic setups are explained. Make sure to review the IBM Tivoli Directory Server 5.2 and z/OS LDAP V1R6 documentation for details, especially when setting up in a production environment.

1. SSL Server Authentication between an IBM Tivoli Directory Server 5.2 and a z/OS LDAP V1R6 Client:
 - a. Enable IBM Tivoli Directory Server 5.2 for SSL Communications:
 - Create a key database using GSKi t key management software that came with the IBM Tivoli Directory Server
 - Transfer myCAcert over to the Linux on zSeries system hosting the ITDS and install it to the server’s key database, using the ITDS 5.2 Administration document.
 - Using the ITDS 5.2 Administration document, create a certificate request for the directory server. The Common Name field in the request should be the name of your server (for example: the IP address).
 - Get the certificate request signed by your CA. For this example, since we are acting as our own CA, transfer the request to the z/OS system where myCA.kdb is located.
 - Sign the certificate request using gskkyman. For example:


```
gskkyman -g -x 365 -cr itdsServer.req -ct itdsServerCert -k /etc/ldap/myCA.kdb
```
 - **itdsServerCert** is now created. Transfer it back to the Linux on zSeries system, and install the server certificate.
 - b. Create a certificate key database for the z/OS LDAP client and install the CA certificate myCAcert in that database:
 - Start gskkyman on z/OS
 - Follow the instructions to create a new key database
For this example: zOSclient.kdb
 - Pick option 7, “*Import a Certificate*”
 - Enter the certificate file name: myCAcert
 - Label it as you wish, we labeled it as myCAcert
 - Hit enter and the certificate is imported

LDAP Server

- From the “*Key Management Menu*” pick option 2, “*Manage Certificates*” then pick myCAcert. Next pick option 2 “*Set Certificate Trust Status*” to make sure that this CA certificate is trusted.
- c. Next, we tested our setup. We did this by running an ldapsearch command, from z/OS against the IBM Tivoli Directory Server 5.2, with the following options:

```
ldaldapsearch -h <host name> -p <secure port> -Z -K <client kdb> -P <client key database password> -b <search string> <filter>
```

For example:

```
ldapsearch -h idsBox -p 636 -Z -K /u/test/keys/z0Sclient.kdb -P secret -b "cn=John Doe, o=Your Company, c=US" objectclass=*
```

2. SSL Server Authentication between a z/OS V1R6 LDAP Server and an IBM Tivoli Directory Server 5.2 Client

- a. Create a certificate key database for IBM Tivoli Directory Server 5.2 client, using GSKit.
- b. Transfer the CA certificate myCAcert, which we created, to the Linux on zSeries system. Install the CA certificate to the client key database using GSKit.
- c. Create a certificate key database, import the CA certificate into it and create a new certificate request for the z/OS LDAP server using gskkyman:
 - Start gskkyman.
 - Create a new key database using the instructions. For example: z0Sserver.kdb
 - Pick option 7, “*Import a Certificate*”.
 - Import myCAcert (the CA certificate we created earlier).
 - Pick option 4, “*Create New Certificate Request*”
 - Follow the instructions to create a new certificate request for your z/OS LDAP server. For this example we used the following values:

```
Enter certificate type (press ENTER to return to menu): 1
Enter request file name (press ENTER to return to menu): z0Sserver.req
Enter label (press ENTER to return to menu): z0SserverCert
Enter subject name for certificate
Common name (required): cn=<server's IP goes here>
Organizational unit (optional): my0u
Organization (required): my0rg
City/Locality (optional): Pok
State/Province (optional): NY
Country/Region (2 characters - required): US
```

Note:

The “Common name” value above must be in that format: cn=server name or IP.

- d. Exit out of gskkyman. Now you must sign the certificate request using gskkyman and the CA certificate we created earlier:

```
gskkyman -g -x 365 -cr z0Sserver.req -ct z0SserverCert -k /u/test/keys/myCA.kdb
```

Now you have a certificate for your server named z0SserverCert.

- e. Install the certificate into your server key database. Start gskkyman, open z0Sserver.kdb, pick option 5, “*Receive requested certificate or a renewal certificate*”, enter filename when requested to, z0SserverCert and hit enter.
- f. Enable z/OS V1R6 LDAP Server for SSL communications following the instructions in “Integrated Security Services LDAP Server Administration and Use” document, use z0Sserver.kdb as your server’s certificate key database.

- g. Next, we tested our setup. We did this by running an `ldapsearch` command, from the Linux on zSeries system against the z/OS LDAP server, with the following options:

```
ldapsearch -h <host> -p <secure port> -P <client's key db>
-b <search string> <filter>
```

For example:

```
ldapsearch -h z0Saddress -p 636 -P /home/test/keys/itdsClient.kdb -b "cn=John Doe,
o=Your Company,c=US" objectclass=*
```

3. SSL Server and Client authentication between an IBM Tivoli Directory Server 5.2 and a z/OS LDAP V1R6 Client

- a. Make sure you have SSL Server Authentication setup between the IBM Tivoli Directory Server 5.2 and z/OS LDAP Client. If not, follow the instructions in 1 on page 193.

- b. Now you should have:

```
Client key database on z/OS system: z0ScIient.kdb
Server key database on the zLinux system: zLinuxServer.kdb (for example)"
Both databases should contain a copy of: myCAcert CA certificate
```

- c. Next, we created a client certificate (for z/OS LDAP Client) request and signed it by way of the CA certificate we created earlier. In this example we created a client certificate for “cn=John Doe,o=Your Company,c=US” entry in the IBM Tivoli Directory Server.

- Start `gskkyman`
- Open `z0ScIient.kdb`
- Pick option 4, “*Create New Certificate Request*”
- Follow the instructions to create a new certificate request for “cn=John Doe,o=Your Company,c=US”. For this example we used the following values:

```
Enter certificate type (press ENTER to return to menu): 1
Enter request file name (press ENTER to return to menu): JohnDoe.req
Enter label (press ENTER to return to menu): JohnDoeCert
Enter subject name for certificate
Common name (required): John Doe
Organizational unit (optional):
  Organization (required): Your Company
  City/Locality (optional):
  State/Province (optional):
  Country/Region (2 characters - required): US
```

Once the certificate request is created, sign the request:

```
gskkyman -g -x 365 -cr JohnDoe.req -ct JohnDoeCert -k /u/test/keys/myCA.kdb
```

Now you have a certificate named `JohnDoeCert`.

- d. Import the client certificate `JohnDoeCert` into your client key database `z0ScIient.kdb`:

- Start `gskkyman`
- Open `z0ScIient.kdb` using the instructions:
- Pick option 7, “*Import a Certificate*”
- Import `JohnDoeCert`

- e. Next, we tested our setup:

```
ldapsearch -h itdsServer -p 636 -Z -S EXTERNAL -N JohnDoeCert -K
/u/test/keys/client.kdb -P secret -b "cn=John Doe,o=Your Company,c=US" objectclass=*
```

4. SSL Server and Client authentication between a z/OS V1R6 LDAP Server and a IBM Tivoli Directory Server 5.2 Client

- a. Make sure you have SSL Server Authentication setup between the z/OS LDAP Server and IBM Tivoli Directory Server 5.2. If not, follow the instructions in 2 on page 194.

LDAP Server

b. Now you should have:

Client key databases on Linux on zSeries system: `itdsClient.kdb`
Server key database on z/OS system: `zOSserver.kdb`
Both databases should contain a copy of: `myCAcert` CA certificate

c. Next, we created a client certificate request (for ITDS LDAP Client) using GSKit.

This example uses a client certificate request called `JohnDoe.req` for the entry `"cn=John Doe,o=Your Company,c=US"`.

Once the certificate request is created, transfer the request to z/OS. Sign the request: `gskkyman -g -x 365 -cr JohnDoe.req -ct JohnDoeCert -k /u/test/keys/myCA.kdb`

We now had a certificate named `JohnDoeCert`, and transferred it back to our Linux on zSeries system.

d. Install the client certificate `JohnDoeCert` into your client key database `itdsClient.kdb` using GSKit.

e. Next, we tested our setup.

Caution: Do not use the `-D` and `-w` options with client authentication, as the bind operation will use the authentication credentials specified with `--D` and `-w` instead of the certificate credentials desired. For example, using the certificate, certificate key database names ... etc from the example above:

```
ldapsearch -h zOSaddress -p 636 -Z -K /home/test/keys/itdsClient.kdb -N  
JohnDoeCert -P secret -b "cn=John Doe, o=Your Company, c=US" "objectclass=*"
```

LDAP Server enhancements in z/OS V1R6

The following topics describe some of the new LDAP Server functions in z/OS V1R6 that we implemented and tested.

- “LDAP migration to z/OS V1R6”
- “Setting up a peer-to-peer replication network between an IBM Tivoli Directory Server 5.2 and a z/OS LDAP Server” on page 197
- “Using DB2 restart/recovery function” on page 203
- “Using alias support” on page 205
- “Using the enhanced LDAP configuration utility (LDAPCNF)” on page 206
- “Using change logging with TDBM” on page 207

LDAP migration to z/OS V1R6

Accessing SYS1.SIEALNKE: All Integrated Security Server products are now placing their load modules in `SYS1.SIEALNKE` instead of maintaining their own load module data set. This is a new data set for the Integrated Security Server products. We used *z/OS Integrated Security Services LDAP Server Administration and Use* and *z/OS Migration* to migrate to this new level and use this new data set.

Prior to starting any LDAP servers, verify that the `SYS1.SIEALNKE` data set is in the `LNKLST` concatenation. If it is not in link list, then you must use `STEPLIB` to locate the data set. When `SYS1.SIEALNKE` is not in the `LNKLST` concatenation, the LDAP server will not start and the following error is seen in the JES log.

```
IEF403I LDAPSRV - STARTED - TIME=10.17.38  
CSV003I REQUESTED MODULE GLDSLAPD NOT FOUND  
CSV028I ABEND806-04 JOBNAME=LDAPSRV STEPNAME=LDAPSRV  
IEA995I SYMPTOM DUMP OUTPUT  
SYSTEM COMPLETION CODE=806 REASON CODE=00000004
```

Using enhanced dynamic, nested, or expanded static group data: In order to use the enhanced support for static, dynamic, and nested groups of users, your DB_Version level must be 3.0 or higher. When we first brought up our z/OS V1R6 LDAP server, our DB_Version was below the required level. We received the following message from the LDAP server:

GLD3148I Dynamic, nested, or expanded static group data is present in the TDBM backend but ignored since the DB_VERSION is not 3.0 or greater below suffixes: suffixes

To resolve this problem, you must update the DB_Version level from SPUFI (SQL Processor Using File Input) with the following statement

```
UPDATE dbuserid.DIR_MISC SET DB_VERSION='3.0'
```

dbuserid is the z/OS user ID that will be the owner of the DB2 tables, a value assigned during initial LDAP backend setup. For example if you defined the *dbuserid* as LDAPSRV when you set up the backend, the SPUFI DB_Version update statement would be:

```
UPDATE LDAPSRV.DIR_MISC SET DB_VERSION='3.0'
```

Once you have updated the DB_Version level to 3.0 or higher, you can use the enhanced dynamic, nested, or static group data. See *z/OS Integrated Security Services LDAP Server Administration and Use* for more information.

Setting up a peer-to-peer replication network between an IBM Tivoli Directory Server 5.2 and a z/OS LDAP Server

The procedures documented here are intended to give an LDAP administrator a set of instructions on how to set up a peer-to-peer replication network between two IBM directory servers, IBM Tivoli Directory Server 5.2 and z/OS LDAP Server on z/OS V1R6.

The procedures assume that you have installed and can use the Web Administration Tool for the IBM Tivoli Directory Server. See the *IBM Tivoli Directory Server Version 5.2 Installation Guide* for information about installing the Web Administration Tool. Another assumption is that you have decided which suffix to replicate and that the entries under that suffix are loaded in both directories.

There are two configuration options presented here. For each option, the procedure starts by creating a master/slave replication network and then promoting that to a peer-to-peer replication network.

Configuration Option 1

This option shows you how to setup a master/slave replication network with IBM Tivoli Directory Server 5.2 as the MASTER and z/OS LDAP Server on z/OS V1R6 as the SLAVE.

PART 1 consists of the following steps:

Creating the Master Server: The servers must be running to perform this task.

This task designates an entry as the root of an independently replicated subtree and creates an **ibm-replicasubentry** representing this server as the single master for the subtree. To create a replicated subtree, you must designate the subtree that you want the server to replicate.

1. Use the Web Administration Tool to log on to the master server.
2. Expand the Replication management category in the navigation area of the Web Administration Tool and click **Manage topology**.

LDAP Server

3. Click **Add subtree**.
4. Enter the DN of the subtree that you want to replicate or click **Browse** to expand the entries to select the entry that is to be the root of the subtree.

Note: If you are not using a suffix, there are other requirements. See *IBM Tivoli Directory Server Version 5.2 Administration Guide*.

5. The master server referral URL is displayed in the form of an LDAP URL. For example,

```
ldap://<myservername>.<mylocation>.<mycompany>.com
```

Note: The master server referral URL is optional. It is used only:

- If the server contains (or will contain) any read-only subtrees.
- To define a referral URL that is returned for updates to any read-only subtree on the server.

6. Click **OK**.
7. The new server is displayed on the Manage topology panel under the heading **Replicated subtrees**.

Creating Credentials: Credentials identify the method and required information, such as a DN and password, which the supplier uses in binding to the consumer.

1. If you have not already done so, use the Web Administration Tool to log on to the master server.
2. Expand the Replication management category in the navigation area of the Web Administration Tool and click **Manage credentials**.
3. Select **cn=replication,cn=IBMpolicies** to store the credentials from the list of subtrees.
4. Click **Add**.
5. Enter the name for the credentials you are creating. For example, **mycreds**, **cn=** is already filled in the field for you.
6. Select **Simple bind** as the type of authentication and click **Next**.
 - Enter the DN that the server uses to bind to the replica. For example, **cn=any**.

Note: This DN cannot be the same as your server administration DN.

- Enter the password the server uses when it binds to the replica. For example, **secret**.
- Enter the password again to confirm that there are no typographical errors.
- If you want, enter a brief description of the credentials
- Click **Finish**.

Note: You might want to record the credential's bind DN and password for future reference. You will need this password when you create the replica agreement.

Creating a replica server: The servers must be running to perform this task.

1. If you have not already done so, use the Web Administration Tool to log on to the master server.
2. Expand the Replication management category in the navigation area of the Web Administration Tool and click **Manage topology**.
3. Select the subtree that you want to replicate and click **Show topology**.

4. Click the arrow next to the **Replication topology** selection to expand the list of supplier servers.
5. Select the supplier server and click **Add replica**.
6. On the **Server** tab of the **Add replica** window:
 - Enter the host name of the replica server. Do not change the default non-SSL port (389).
 - Leave the **Enable SSL** check box unchecked.
 - Enter the replica name or leave this field blank to use the host name.
 - Enter the replica ID. If the server on which you are creating the replica is running, click **Get replica ID** to automatically fill this field. For a replica, which is a z/OS LDAP server, this section will be filled as **UNKNOWN**. Enter a description of the replica server.
7. Click the **Additional** tab.
 - Specify the credentials that the replica uses to communicate with the master:
 - Click **Select**.
 - Click the radio button next to **cn=replication,cn=IBM policies**.
 - Click **Show credentials**.
 - Expand the list of credentials and select **mycreds**.
 - Click **OK**.

See “Creating Credentials” on page 198 for additional information on agreement credentials.
 - Keep the **Specify a replication schedule or enter DN (optional)** set to **None**. This sets the default as immediate replication.
 - Do not deselect any capabilities.
 - Click **OK** to create the replica. A message is displayed noting that additional actions must be taken.
 - Click **OK**.
8. Next, the supplier information must be added to the replica. Open the **slapd.conf** configuration file of the z/OS LDAP server. Find the TDBM backend definitions and add the following configuration file options under the **suffix** that is to be replicated: **masterserver, masterserverdn, masterserverpw**.

Example:

```
masterserver ldap://<MasterServerIP>:<MasterServerPort>/
masterserverdn cn=any
masterserverpw password
```

For **masterserverdn** and **masterserverpw** use the credentials you created in “Creating Credentials” on page 198.

Restart the replica.

Starting replication: The replica is in a suspended state and no replication is occurring. After you have finished setting up your replication topology, on the master you must:

1. If you have not already done so, use the Web Administration Tool to log on to the master server.
2. Expand the **Replication management** category in the navigation area of the Web Administration Tool and click **Manage queues**.
3. Select the new replica.
4. Click **Suspend/resume** to start receiving replication updates for that server.

Master/slave replication setup is complete.

PART 2, in which you promote the master/slave replication network to a peer-to-peer replication network, consists of the following steps:

Changing the master to a peer: Refer to the **current** master as peer1 and the **current** replica as peer2. To define peer2 to peer1, we add peer2's definition to peer1's configuration file **ibmslapd.conf**.

For example:

```
dn: cn=Master Server,cn=Configuration
cn: Master Server
ibm-slapdMasterDN: cn=peer
ibm-slapdMasterPW: secret
objectclass: ibm-slapdReplication
objectclass: top
```

Promoting the replica to a peer: Stop peer2.

Open the **slapd.conf** configuration file for peer2 and delete **masterserver**, **masterserverDN**, and **masterserverPW** configuration file options.

Restart peer2.

Next, define peer1 to peer2. Create an LDIF file as in the example below and add it to peer2's directory, using the **ldapadd** utility.

For example:

```
dn: cn=myReplica,o=Your Company,c=US
objectclass: top
objectclass: replicaObject
cn: myReplica
replicaHost: <ip address>
replicaBindDn: cn=peer
replicaCredentials: secret
```

Stop peer2.

Open peer2's **slapd.conf** configuration file, find the TDBM backend definitions and add **peerServerDN** **peerServerPW** configuration file options under the **suffix** that is being replicated.

Example:

```
peerServerDN cn=peer
peerServerPW secret
```

Starting peer-to-peer replication: Restart servers peer1 and peer2. Peer-to-peer replication network setup between an IBM Tivoli Directory Server 5.2 and a z/OS LDAP Server on z/OS V1R6 is complete.

Configuration Option 2

This option shows you how to setup a master/slave replication network with z/OS LDAP Server on z/OS V1R6 as the MASTER and IBM Tivoli Directory Server 5.2 as the SLAVE.

PART 1 consists of the following steps:

Setting up IBM Tivoli Directory Server 5.2 as the Slave: Before setting up IBM Tivoli Directory Server 5.2 as the slave, we set it up as the master.

Follow the instructions in “Configuration Option 1” on page 197 up until 8 on page 199 to set up IBM Tivoli Directory Server 5.2 as the master and z/OS LDAP Server on z/OS V1R6 as the slave.

Next, delete the replication agreement that was just built between the master and the slave. That action leaves behind some information in the IBM Tivoli Directory Server that we will use later on.

1. Use the Web Administration Tool to log on to the master server.
2. Expand the **Replication management** category in the navigation area of the Web Administration Tool and click **Manage topology**.
3. Select the subtree that you picked to replicate earlier and click **Show topology**.
4. Click the arrow next to the **Replication topology** selection to expand the list of all the servers within the replication network. Keep doing this until both servers are listed.
5. Delete the replica server from the topology. (It is the bottom one in the list.)

Follow the directions below to look into the IBM Tivoli Directory Server 5.2 under “ibm-replicaGroup=default,<subtreedn>”. You should find an ibm-replicaSubEntry object with the attribute ibm-replicationServerIsMaster=TRUE. Change this attribute value to FALSE.

1. Use the Web Administration Tool to log on to the master server.
2. Expand the **Directory management** category in the navigation area of the Web Administration Tool and click **Manage entries**.
3. Select the subtree that is being replicated and then click **Find**.
4. Pick **ibm-replicaSubEntry** objectclass from the **Find Entries with Following Objectclasses** drop down menu and click **OK**.
5. Click **Edit Attributes**.
6. Pick **FALSE** under **ibm-replicationServerIsMaster** section.
7. Click **OK, OK, and CANCEL** to go back to the **Manage entries** window.

Next, we need to define the new master server, z/OS V1R6 LDAP Server, to its replica, IBM Tivoli Directory Server 5.2. We do this by adding a description of the master to IBM Tivoli Directory Server’s **ibmslapd.conf** configuration file. Use the description below as an example:

```
dn: cn=Master Server,cn=Configuration
cn: Master Server
ibm-slapdMasterDN: cn=peer
ibm-slapdMasterPW: secret
ibm-slapdMasterReferral: ldap://<MasterServerIP>:<MasterServerPort>/
objectclass: ibm-slapdReplication
objectclass: ibm-slapdConfigEntry
objectclass: top
```

Finally, go back to the Web Administration Tool and click on **Replication Management**, then click on **Manage Topology**. Pick the subtree that is being replicated and click on **Edit Subtree**. Change the current referral address to the master server’s (z/OS V1R6 LDAP Server) address and click **OK**.

Setting up z/OS V1R6 LDAP Server as the Master: Now, define the replica to its master. Create an LDIF file similar to the example below, which includes a `replicaObject` representing the replica server. Then, add the LDIF file to the master's directory, using the `ldapadd` utility.

```
dn: cn=myReplica,o=Your Company,c=US
objectclass: top
objectclass: replicaObject
cn: myReplica
replicaHost: <ip address>
replicaBindDn: cn=peer
replicaCredentials: secret
```

Starting replication: Restart both servers.

The master/slave replication between an IBM Tivoli Directory Server 5.2 and a z/OS LDAP Server on z/OS V1R6 is complete.

PART 2, in which we promote the master/slave replication network to a peer-to-peer replication network, consists of the following steps:

Changing the master to a peer: Stop the master server.

Open the `slapd.conf` configuration file. Find the TDBM backend definitions. Add `peerServerDN` and `peerServerPW` configuration file options under the `suffix` that is to be replicated.

Example:

```
peerServerDN cn=peer
peerServerPW secret
```

Changing the replica to a peer:

1. Use the Web Administration Tool to log on to the master server.
2. Expand the Replication management category in the navigation area of the Web Administration Tool and click **Manage topology**.
3. Select the subtree that you want to replicate and click **Edit subtree**.
4. Scroll right and click on **Make server a master**. A message will pop up. Click **OK**, click **OK** on the next screen.
5. Click **Show topology**.
6. Click the arrow next to the **Replication topology** selection to expand the list of supplier servers. Select the supplier server and click **Add replica**.
7. On the **Server** tab of the **Add replica** window:
 - Enter the host name of the replica server. Do not change the default non-SSL port (389).
 - Leave the **Enable SSL** check box unchecked.
 - Enter the replica name or leave this field blank to use the host name.
 - Enter the replica ID. If the server on which you are creating the replica is running, click **Get replica ID** to automatically fill this field. For a replica, which is a z/OS LDAP server, this section will be filled as **UNKNOWN**. Enter a description of the replica server.
8. Click the **Additional** tab.
 - Specify the credentials that the replica uses to communicate with the master:
 - Click **Select**.
 - Click the radio button next to **cn=replication,cn=IBM policies**.

- Click **Show credentials**.
- Expand the list of credentials and select **mycreds**.
- Click **OK**.

See “Creating Credentials” on page 198 for additional information on agreement credentials.

- Keep the **Specify a replication schedule or enter DN (optional)** set to **None**. This sets the default as immediate replication.
- Do not deselect any capabilities.
- Click **OK** to create the replica. A message is displayed noting that additional actions must be taken.
- Click **OK**.

Starting peer-to-peer replication: Restart IBM Tivoli Directory Server 5.2 and then restart z/OS LDAP Server.

Peer-to-peer replication network is complete.

Reference information

We used the following documentation to set up these procedures:

- *z/OS Integrated Security Services LDAP Server Administration and Use*
- *A Simplified Approach to IBM Tivoli Directory Server V5.2 Replication*
- *IBM Tivoli Directory Server Administration Guide*

Using DB2 restart/recovery function

DB2 Restart/Recovery is a new feature for the z/OS V1R6 LDAP server that allows a TDBM and/or GDBM configured LDAP server to remain up and running even if DB2 shuts down. Then, once DB2 is restored, the LDAP server can function as normal. In the past, you had to restart the LDAP server in order to reconnect to DB2 once the connection was lost.

We used the following setup and tested DB2 restart/recovery:

1. Installed the fix for APAR PQ87724, which is required for this function.
-
2. Updated the slapd.conf file as follows:

```
# -----
database tdbm GLDBTDBM
suffix "o=xxx"
suffix "o=xxx"
servername xxxxxxxxx
dbuserid xxxxxx
databasename xxxxxx
dsnaoini GLD.CNFOUT.xx(DSNAOINI)
attroverflowsize 500
pwEncryption none
schemaReplaceByValue on
extendedgroupsearching on
db2terminate restore

# GDBM-specific CONFIGURATION SETTINGS
# -----
# -----
database gdbm GLDBGDBM
servername xxxxxxxxx
dbuserid xxxxxx
dsnaoini GLD.CNFOUT.xxx(DSNAOINI)
```

```
changeLogging on
changeLogMaxEntries 1000
changeLogMaxAge 86400
attroverflowsz 500
schemaReplaceByValue on
db2terminate restore
```

Note that db2terminate restore is the default, so that if you do not specify a value, you'll get restore. If you specify db2terminate terminate, the LDAP server will shut down when DB2 shuts down.

3. We tested DB2 restart/recovery function by bringing DB2 down and up again, and verifying that LDAP stayed up throughout:

- After we brought DB2 down, the system issued the following LDAP server message:

```
GLD0252E DB2 termination detected, database access unavailable.
```

To verify that DB2 was down, we issued an ldapsearch command:

```
ldapsearch -h <hostname> -b "o=IBM" "objectclass=*
```

The system returned a message saying that no such object was present, verifying that DB2 was down.

- We brought DB2 up again, and received the following LDAP server message:

```
GLD0253I DB2 restart detected, database access available.
```

Again, we issued the ldapsearch command, this time to verify that DB2 was up:

```
ldapsearch -h <hostname> -b "o=IBM" "objectclass=*
```

The system displayed the output for the search query.

Migrating to DB2 V8

When we migrated to DB2 V8 we encountered the following problems with LDAP.

Module DSNAOCLI was not found in an authorized library

The LDAP server would not start. The following message was seen in the servers JES log.

```
CEE3518S The module DSNAOCLI was not found in an authorized library.
          From entry point DBXAllocEnv at compile unit offset +000000D0 at
          entry offset +000000D0 at address 277556A0.
CSV042I REQUESTED MODULE DSNAOCLI NOT ACCESSED. THE MODULE IS NOT PROGRAM CONTROLLED.
ICH422I THE ENVIRONMENT CANNOT BECOME UNCONTROLLED.
BXP014I ENVIRONMENT MUST REMAIN CONTROLLED FOR SERVER (BPX.SERVER) PROCESSING.
```

DSNAOCLI is a DB2 module that is in dataset DB2.DB2810.SDSNLOAD. To resolve the above problem we program controlled the dataset with the following commands.

```
ralter program * addmem('DB2.DB2810.SDSNLOAD'/'*****/NOPADCHK)
SETROPTS WHEN(PROGRAM) REFRESH
```

Plans needed to be rebound using SQLERROR(CONTINUE)

After program controlling the dataset the server started but the TDBM backend failed to configure. We encountered the following errors:

```

| GLD0154E Error code -1 from odbc string: "SQLConnect " USIBMT6PETDB2 .
| GLD0155E ODBC error, SQL data is: native return code=-805, SQL state=51002,
| SQL message={DB2 FOR OS/390}{ODBC DRIVER}
| DSNT408I SQLCODE = -805, ERROR: DBRM OR PACKAGE NAME USIBMT6PETDB2..DSNCLIC1.-
| 177B36231891080D NOT FOUND IN PLAN DSNACJA. REASON 03

```

There were problems binding the plan that the DB2 System Programmers had to resolve. The problem was the plans needed to be rebound using SQLERROR(CONTINUE) as per holddata for PTF UQ87683.

Once these were resolved and the Plan was bound LDAP worked successfully.

Using alias support

Alias support provides a means for a TDBM directory entry to point to another entry in the same TDBM directory. If a distinguished name encountered during a search operation with dereferencing contains an alias, the alias is replaced by the value it points to and search continues using the new distinguished name. This support is designed to allow a user to make directory information available even if the entry is moved. It allows the user to point to a well-known name that would always lead to the entry.

We did the following to setup and test alias support:

1. Created an Idif file to exercise alias support. New file **alias.ldif** contains the following:

```

dn: cn=js, o=IBM
objectclass:person
objectclass:aliasObject
sn: Smith
aliasedobjectname: cn=Joe Smith, ou=My Team, ou=Test Team, o=IBM

```

2. Next we loaded the alias.ldif file into the TDBM database with the following ldapmodify command:

```
ldapmodify -h <hostname> -D "cn=LDAP Administrator" -w xxxxx -f /sysname/etc/ldap/alias.ldif
```

3. To make sure that alias.ldif was properly loaded, we issued the following ldapsearch command:

```
ldapsearch -h <hostname> -b "o=js, o=IBM" "objectclass=*
```

This command displayed the following output, showing the new alias entry, which showed that alias.ldif was loaded properly:

```

dn: cn=js, o=IBM
objectclass:person
objectclass:aliasObject
sn: Smith
aliasedobjectname: cn =Joe Smith, ou=My Team, ou=Test Team, o=IBM

```

4. We tested alias support by issuing the following ldapsearch command with new parameter **-a always** that specifies that aliases are always dereferenced:

```
ldapsearch -h <hostname> -a always -b "cn=js, o=IBM" "objectclass=*
```

This command displays the actual entry that the alias represents:

```
cn=Joe Smith, ou=My Team, ou=Test Team, o=IBM
```

See *z/OS Integrated Security Services LDAP Client Programming* for information on the `-a deref` parameter.

Using the enhanced LDAP configuration utility (LDAPCNF)

The LDAP configuration utility, `ldapcnf`, has been enhanced so that configuring TDBM is no longer required and to support configuring the change log. We tested the enhanced LDAPCNF utility using information from *z/OS Integrated Security Services LDAP Server Administration and Use*. We did the following to setup and test `ldapcnf`:

1. Created a new LDAP server for testing `ldapcnf` by doing the following:
 - Copied the following files from `/usr/lpp/ldap/etc` to `/sysname/etc/ldap`:

 `ldap.profile`
 `ldap.slapd.profile`
 `ldap.db2.profile`
 `ldap.racf.profile`
 `slapd.conf`
 - Next, we updated the profile files specific to our new LDAP server. We made the following changes to profile files:
 - **ldap.profile** and **ldap.slapd.profile**- We made system specific changes, see *z/OS Integrated Security Services LDAP Server Administration and Use* for information.
 - **ldap.db2.profile** - We enabled the LDAP server for TDBM and GDBM. You can choose to configure one or both.
 - **ldap.racf.profile** - We made system specific changes, see *z/OS Integrated Security Services LDAP Server Administration and Use* for information..

-
2. Started the LDAP configuration utility, `ldapcnf`, using the following command:

```
ldapcnf -i ldap.profile
```

The `ldapcnf` utility ran successfully with a return code of 0 and the utility created the following:

- JCL jobs
 - SLAPDCNF (LDAP server configuration file)
 - SLAPDENV (LDAP server environment variable file)
 - PROG member needed for APF authorization
 - Procedure needed to start the LDAP server
 - DSNAOINI configuration file for DB2 CLI
 - SPUFI DB2 SQL Statements for TDBM and GDBM
-
3. Next we copied the LDAP started task procedure to the procedure library for our new LDAP server and copied the PROGxx member to the target system's PARMLIB.
-
4. We then submitted the APF member and DBCLI member following JCL jobs. Our DB2 administrators submitted the DBSPUFI members for TDBM and GDBM using the DB2 SPUFI tool.
-
5. Finally, we started up the LDAP server as follows:

```
s user_id
```

We received the following message:

```
GLD0122I Slapd is ready for requests.
```

The LDAP Server was running with both TDBM and GDBM configured successfully. We successfully ran a variety of LDAP commands to test this server.

Using change logging with TDBM

For the z/OS V1R6 LDAP server, change logging has been enabled for changes made to entries in the TDBM backend. When a change is made to an entry in the TDBM backend, a record of the change will be created and stored in the GDBM backend.

To set up and test TDBM change logging, we did the following:

1. Because this support includes a new backend section (GDBM) in the configuration file, we first enabled the GDBM back end for the LDAP server in the configuration file:

```
# GDBM-specific CONFIGURATION SETTINGS
# -----
# -----
database gdbm GLDBGDBM
servername xxxxxxx
dbuserid xxxxxx
#databasename xxxxxx
dsnaoini GLD.CNFOUT.xxx(DSNAOINI)
changeLogging on
changeLogMaxEntries 1000
changeLogMaxAge 86400
attroverflowsize 500
schemaReplaceByValue on
```

Once the GDBM backend is configured, the change logging support is automatically enabled for the corresponding TDBM back end. Note that **changeLogging on** is the default setting.

2. After this setup step, the slapd.conf file looks as follows (the bold sections show requirements for change logging):

```
# * This file is the LDAP Server configuration file for z/OS.
# *****/
#global section

timelimit      3600
sizelimit      500

adminDN "cn=LDAP Administrator"
adminPW xxxxxx

listen ldap://:389
listen ldap://:pc
# TDBM-specific CONFIGURATION SETTINGS
# -----
# -----
changeLogging on
database tdbm GLDBTDBM
```

```

suffix "o=IBM"
suffix "o=Your Company"
servername xxxxxxxxxx
dbuserid xxxxxx
databasename xxxxxxx
dsnaoini GLD.CNFOUT.xx(DSNAOINI)
attroverflowsize 500
pwEncryption none
schemaReplaceByValue on
extendedgroupsearching on
# GDBM-specific CONFIGURATION SETTINGS
# -----
# -----
database gdbm GLDBGDBM
servername xxxxxxxxxx
dbuserid xxxxxx
dsnaoini GLD.CNFOUT.xx(DSNAOINI)
changeLogging on
changeLogMaxEntries 1000
changeLogMaxAge 86400
attroverflowsize 500
schemaReplaceByValue on
#####
# sdbm database definitions
#####
database sdbm GLDBSDBM
suffix "sysplex=xxxxxxx,0=IBM,C=US"

```

3. Finally, we restarted the LDAP server. SLAPDOUT displays the following, showing that ChangeLogging is enabled:

```

GLD0244I Change logging is enabled
Logging started status (0 = off, 1 = on): 1
Limit in seconds on age of change log entries (0 = no limit): 86400
Limit on the number of change log entries (0 = no limit): 1000
Current number of change log entries: 0
First change number in use: 0
Last change number in use: 0
GLD3151I The backend containing the following suffix is participating in change logging: 'CN=CHANGELOG'.
GLD3151I The backend containing the following suffix is participating in change logging: 'o=IBM'.
GLD0202I Program Call communication is active.
GLD0122I Slapd is ready for requests.

```

4. Now we're ready to start testing the TDBM change logging. First, we created an ldif file, test.ldif, that would make a change to an entry in the TDBM backend:

```

dn: cn=John Doe, ou=My Team, ou=Test Team, o=IBM
changetype:modify
replace:x
title:ICSF

```

5. We then issued the following ldapmodify command to make the changes in the ldif file:

```

ldapmodify -h <hostname> -D "cn=LDAP Administrator" -w xxxxx -f test.ldif

```

The modifications were made successfully.

6. I then issued an ldapsearch against the change log to verify that a record was created for the change:

```

ldapsearch -h <hostname> -b "cn=changelog" "objectclass=*"

```

The following output was displayed, showing that TDBM change logging was working successfully:

```
cn=changelog
objectclass=top
objectclass=container
cn=changelog

CHANGENUMBER=1201,CN=CHANGELOG
objectclass=CHANGELOGENTRY
objectclass=IBM-CHANGELOG
objectclass=TOP
changenumber=1201
targetdn=cn=John Doe, ou=My Team, ou=Test Team, o=IBM

changetime=20040701185439.759260Z
changetype=MODIFY
changes=replace:title
title: ICSF
-
add:ibm-entryuuid
ibm-entryuuid: B95FA000-5DEF-10E4-B0B9-40208401B52A
-
ibm-changeinitiatorsname=CN=LDAP ADMINISTRATOR
```

TDBM change logging works successfully.

LDAP Server

Chapter 14. Using Kerberos (Network Authentication Service)

Setting up a Kerberos peer trust relationship between z/OS and Windows 2000

We had a request from our colleagues in z/OS LDAP development to test the z/OS LDAP client with the Microsoft Windows 2000 Active Directory service using a Kerberos authentication bind. This was to help validate a fix they were working on involving the use of Kerberos authentication to bind to IBM Directory Server.

We needed to create a Kerberos realm that included Kerberos servers on z/OS and a Windows platform. Another test team had already enabled a Windows 2000 server for Kerberos authentication and we already had a Kerberos server running on z/OS. We now needed to create a peer trust relationship between these two Kerberos servers.

Since the Windows 2000 server was set up by another test team, we don't have a lot of details to share. However, we intend to continue our testing with Kerberos peer relationships and transitive trust relationships on other platforms and from other suppliers. We'll have more to say about how to enable such scenarios at that time. For now, we will focus on what we did from the z/OS perspective to set up the peer trust relationship and how we validated our setup.

We used the following documentation to help us set up the z/OS portion of the peer trust relationship:

- *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

Enabling the peer trust relationship on z/OS

There are two main areas that need to be configured to enable the peer trust relationship for Kerberos authentication from the z/OS perspective: the `krb5.conf` configuration file and RACF. We configured these accordingly, as described below, to define a new Kerberos realm. (For more information, see the appendix containing sample Kerberos configurations in *z/OS Integrated Security Services Network Authentication Service Administration*.)

Defining the Windows 2000 realm to the Kerberos server on z/OS

We did the following to update the `/etc/skrb/krb5.conf` configuration file:

1. Under the `[Realms]` section, we defined the Windows 2000 server as follows:

```
KERBEROS.XXX.YYY.IBM.COM = {  
    kdc = kerb2000.kerberos.xxx.yyy.ibm.com:88  
    kpasswd_server = kerb2000.kerberos.xxx.yyy.ibm.com:464  
}
```

2. Under the `[domain_realm]` section, we defined the Windows 2000 server as follows:

```
.kerberos.xxx.yyy.ibm.com = KERBEROS.XXX.YYY.IBM.COM
```

Kerberos-based peer trust

Defining the cross-realm certification in RACF

We issued the following RACF commands to define the cross-realm certification:

```
RDEFINE REALM /.../KERBEROS.XXX.YYY.IBM.COM/krbtgt/XXX.YYY.IBM.COM KERB(PASSWORD(win2k_pw))  
RDEFINE REALM /.../XXX.YYY.IBM.COM/krbtgt/KERBEROS.XXX.YYY.IBM.COM KERB(PASSWORD(zos_pw))
```

Testing the peer trust relationship

Since the intention for this scenario was to test the z/OS LDAP client, we used the z/OS LDAP client and the Windows 2000 Active Directory service to validate the newly created Kerberos realm and the peer trust relationship.

We did the following to test the peer trust relationship (all commands are issued from the z/OS UNIX shell):

1. We issued the **kinit** command to obtain Kerberos credentials. Because we were trying to obtain credentials from the Windows 2000 server, we issued the **kinit** command using a Windows 2000 Kerberos principal.

Example: `kinit SAM@KERBEROS.XXX.YYY.IBM.COM`

Result: EUVF06017R Enter password:

If everything is set up correctly and you enter the correct password, you will simply return to the command prompt when the **kinit** command successfully completes.

However, if there is a problem with the setup and you are unable to access the Windows 2000 server, you would see the following error message:

```
EUVF06014E Unable to obtain initial credentials.  
Status 0x96c73a9a - Unable to locate security server.
```

Our **kinit** command was successful and we returned to the command prompt.

2. We used the **klist** command to verify that we had the expected credentials.

Example: `klist`

Result: We received the following response, as expected:

```
Ticket cache: FILE:/var/skrb/creds/krbcred_0a3ae270  
Default principal: SAM@KERBEROS.XXX.YYY.IBM.COM  
  
Server: krbtgt/KERBEROS.XXX.YYY.IBM.COM@KERBEROS.XXX.YYY.IBM.COM  
Valid 2004/05/13-13:21:23 to 2004/05/13-23:21:23
```

At this point, we were confident that we had established communication between z/OS and the Kerberos server running on Windows 2000. We could now use other applications that require binding with Kerberos authentication. In this scenario, we used the z/OS LDAP client to search the Windows 2000 Active Directory, as described in the following steps.

3. We issued the **ldapsearch** command to search the Windows 2000 Active Directory.

Example: We entered the following command as a single line from the z/OS UNIX command prompt:

```
Ldapsearch -h ip_address_of_win2k_server -V 3 -S GSSAPI -s base  
-b "CN=Users,DC=kerberos,DC=xxx,DC=yyy,DC=ibm,DC=com" "objectclass=*
```

Result: We received the following response, as expected:

```

CN=Users,DC=kerberos,DC=xxx,DC=yyy,DC=ibm,DC=com
cn=Users
description=Default container for upgraded user accounts
instanceType=4
isCriticalSystemObject=TRUE
distinguishedName=CN=Users,DC=kerberos,DC=xxx,DC=yyy,DC=ibm,DC=com
objectCategory=CN=Container,CN=Schema,CN=Configuration,DC=kerberos,DC=xxx,DC=yyy,DC=ibm,DC=com
objectClass=top
objectClass=container
objectGUID=NOT Printable
name=Users
showInAdvancedViewOnly=FALSE
systemFlags=-1946157056
uSNCreated=1314
uSNChanged=1314
whenChanged=20030506135552.0Z
whenCreated=20030506135552.0Z

```

To validate the peer trust relationship between the Kerberos server on z/OS and the Kerberos server on Windows 2000, we needed to clear out our existing Kerberos credentials (that we had obtained using a Windows 2000 Kerberos principal), obtain new Kerberos credentials using a z/OS Kerberos principal, and then rerun the **ldapsearch** command.

4. We issued the following command to clear out any existing Kerberos credentials:

```
kdestroy
```

5. We issued the **kinit** command to obtain new Kerberos credentials using a z/OS Kerberos principal.

Example: `kinit LDAP/zOS.ibm.com`

6. We reissued the same **ldapsearch** command as before to search the Windows 2000 Active Directory.

Example: We entered the following command as a single line from the z/OS UNIX command prompt:

```
ldapsearch -h ip_address_of_win2k_server -V 3 -S GSSAPI -s base
-b "CN=Users,DC=kerberos,DC=xxx,DC=yyy,DC=ibm,DC=com" "objectclass=*
```

Result: As expected, the response was identical to the one received before.

Network Authentication Service (NAS) enhancements in z/OS V1R6

All Integrated Security Server products are now placing their load modules in SYS1.SIEALNKE instead of maintaining their own load module data set. This is a new data set for the Integrated Security Server products.

We used *z/OS Integrated Security Services Network Authentication Service Administration* and *z/OS Migration* when migrating to this new level and using this new data set.

Accessing SYS1.SIEALNKE

Prior to starting any NAS servers, verify that the SYS1.SIEALNKE data set is link list. If it is not in link list, then you must use STEPLIB to locate the data set.

When SYS1.SIEALNKE is not in link list, the NAS server will not start and the following error is seen in the JES log.

Kerberos-based peer trust

```
IEF403I  SKRBKDC - STARTED - TIME=20.41.45
CSV003I  REQUESTED MODULE EUVFSKDC NOT FOUND
CSV028I  ABEND806-04  JOBNAME=SDRBKDC  STEPNAME=SKRBKDC
IEA995I  SYMPTOM DUMP OUTPUT
SYSTEM COMPLETION CODE=806  REASON CODE=00000004
```

FTP with Kerberos

Our goal was to use an FTP client on Linux (on an Intel® box), obtaining the Kerberos credentials from a z/OS Kerberos server and then FTP into a z/OS FTP server using those credentials. This document will identify the steps taken to implement this solution.

Where to find more information

During our testing, we used documentation from several sources, listed below.

- *z/OS Communications Server: IP Configuration Guide*, "Chapter 11. Transferring files using FTP" section titled "Steps for customizing the FTP server for Kerberos"
- *z/OS Security Server RACF Command Language Reference*
- *Communications Server for z/OS V1R2 TCP/IP Implementation Guide Volume 7: Security SG24-6840-00*, section 11.1.3 "FTP using Kerberos".

FTP server enablement for Kerberos

The following sections describe how we set up our FTP server enablement for Kerberos.

Assigning service principals

Initially it was thought that both an FTP and a host service principal were required. After further investigation and testing it was determined that the host service principal is not needed. We needed to create the ftp service principal against a RACF id. Although any id could be used we decided that to keep it simple we would assign the service principal to an id with the same name.

Adding the FTP service principal:

- Add the RACF userid FTP on our systems. (There was already a RACF userid FTP on our systems.) If the FTP RACF userid did not already exist, the following command would be used to create it:

```
adduser FTP NOPASSWORD DFLTGRP(SYS1) omvs(autouid home('/u/ftp') prog('/bin/sh'))
```
- Add the Kerberos principal once the FTP RACF userid is created. We used the following command:

```
ALTUSER FTP PASSWORD(ftp) NOEXPIRED KERB(KERBNAME(ftp/<hostname>))
```
- Remove password protection from the FTP id, if desired. Because a password was not desired to be assigned to the FTP id the following command was issued to remove it:

```
ALTUSER FTP NOPASSWORD
```
- Ensure that the Kerberos segment was added by using the following command to display the id:

```
LU FTP NORACF KERB
```

The following is the result:

```
USER=FTP
```

```
KERB INFORMATION
```

```

-----
KERBNAME= ftp/<hostname>
KEY VERSION= 001
KEY ENCRYPTION TYPE= DES DES3 DESD

```

The ftp.data file for the FTP Server

The ftp.data file resides in the /etc directory.

Note: The file name default is the same for both the FTP server and client. What designates which one to use is the location of the file. The default location of the server's file is in /etc. Both the name of the file and the location can be changed. The changed name or location must then be in the FTPD startup proc. In our installation we have changed the name of this file to ftps.data (we added the s for server copy) to help us distinguish it from the client copy. The client file location is in the clients 'home' directory.

At a minimum the following must be enabled in the file to enable the FTP server for Kerberos:

```
EXTENSIONS          AUTH_GSSAPI
```

This is all that we added to the ftp.data file thus accepting the defaults for the remaining Kerberos specific variables.

Adding the keytab file

We added a keytab file on z/OS for the FTP service principal.

- Locate the keytab file in the /etc/skrb directory:

```
cd /etc/skrb
```

- Use the list command to show what is currently in the keytab file:

```
keytab list
```

The following will be returned if nothing is currently in the keytab file:

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

- Add the FTP service principal with the following command:

```
keytab add ftp/<hostname>
```

- Enter the principals' password.

You will be prompted for the principals' password. For this example that password is FTP and it must be entered in uppercase. This password was assigned via the RACF ALTUSER command when the FTP principal was created.

- Issue the keytab list again.

The following is what should be displayed when the FTP service principal is present.

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

```
Principal: ftp/<hostname>@<realm>
```

```
Key version: 1
```

```
Key type: 56-bit DES
```

```
Entry timestamp: 2005/02/04-16:21:10
```

```
Principal: ftp/<hostname>@<realm>
```

```
Key version: 1
```

```
Key type: 56-bit DES using key derivation
```

```
Entry timestamp: 2005/02/04-16:21:10
```

```
Principal: ftp/<hostname>@<realm>
```

```
Key version: 1
```

```
Key type: 168-bit DES using key derivation
```

```
Entry timestamp: 2005/02/04-16:21:10
```

Kerberos-based peer trust

Running without a keytab file

An alternative to running with a keytab file is to associate the FTP Kerberos principal to the id under which the FTP started task runs. If the id under which FTP runs happens to be named FTP then the examples above for creating the FTP Kerberos principal would be fine. Otherwise let's say the id which the FTP started task runs under is named FTPD. Issue the following command to create the FTP Kerberos principal and have it associated to that id.

```
ALTUSER FTPD PASSWORD(ftp) NOEXPIRED KERB(KERBNAME(ftp/<hostname>))
```

In this setup the KRB5_SERVER_KEYTAB environment variable must be set. This can be specified directly in the FTP startup proc as listed below:

```
//FTPD EXEC PGM=&MODULE,REGION=4096K,TIME=NOLIMIT,  
// PARM=('POSIX(ON) ALL31(ON) ',  
// 'ENVAR("KRB5_SERVER_KEYTAB=1")/&PARMS')
```

The other alternative to specifying the environment variable directly in the startup proc would be to specify a file where the environment variables are listed:

```
//FTPD EXEC PGM=&MODULE,REGION=4096K,TIME=NOLIMIT,  
// PARM=('POSIX(ON) ALL31(ON) ',  
// 'ENVAR("_CEE_ENVFILE=/etc/ftp.envvars")/&PARMS')
```

Then within the /etc/ftp.envvars file add the following:

```
KRB5_SERVER_KEYTAB=1
```

Configuring a Linux workstation for Kerberos

The following software was used when we configured our Linux workstation for Kerberos.

- SUSE 9.2 Linux workstation
- MIT Kerberos Client

We created a new z/OS userid kerbftp1 to be used with the Linux workstation and eventual workload. We used the following RACF commands for this:

```
adduser kerbftp1 NOPASSWORD DFLTGRP(SYS1) omvs(autouid home('/u/kerbftp1') prog('/bin/sh'))  
ALTUSER kerbftp1 PASSWORD(ftp) NOEXPIRED KERB(KERBNAME(kerbftp1))
```

The following was needed for the Linux workstation Kerberos configuration. The data can be found in the Kerberos configuration file /etc/skrb/krb5.conf

Default Realm

This value is found under the [libdefaults] section as default_realm

Default Domain

This value is found under the [domain_realm] section. It is the left hand side of the equation default.domain = default.realm

KDC Server Address

This value is found under the [realms] section as kdc = ip.address:port under the Default Realm designation.

Creating the ftp.data file for the z/OS client ftp user

We created an ftp.data file in the users home directory. (/u/JOE)

Add at a minimum the following to the ftp.data file:

```
SECURE_MECHANISM GSSAPI
```

Testing FTP with Kerberos

We used the following commands to test the setup from the z client perspective.

First the Kerberos credentials are required.

Issue the kinit command:

```
kinit kerbftp1
```

You will be prompted for the password. Enter FTP Then issue the FTP command:

```
ftp <hostname>
```

You should see the following:

```
Using /u/JOE/ftp.data for local site configuration parameters.
IBM FTP CS V1R6
FTP: using TCPIP
Connecting to: <hostname> <ipaddress> port: <portnumber>.
220-FTPD1 IBM FTP CS V1R6 at <hostname>, 21:51:51 on 2005-02-04.
220 Connection will close if idle for more than 5 minutes.
>>> AUTH GSSAPI
334 Using authentication mechanism GSSAPI
>>> ADAT
235 ADAT=YGgGCSqGSIB3EgECAgIAb1kwV6ADAgEFoQMCAQ+iSzBJoAMCAQGiQgRAjucQzx1Yf
d1fLzoc7CSk1SZCL87moSzVQ+fx1CJ9Z5nu0fRpRP9K0DnxmPENQZj7WsFA/nEL4Gpbw+CI8X/kxw==
Authentication negotiation succeeded
NAME (<hostname>:USER):
JOE
>>> USER JOE
331 Send password please.
PASSWORD:

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

Problems encountered

Following are some of the problems we encountered:

1. The need for either the keytab file or association of the Kerberos principal to the FTP started task id is not defined in *z/OS Communications Server: IP Configuration Guide*.
2. There is no information on the requirement or how to create the FTP service principal in *z/OS Communications Server: IP Configuration Guide*.
3. The requirement to specify a user id and password on the FTP transaction should not happen.

The first two documentation problems will be resolved in the z/OS V1R7 level of the books. The third problem has been defined to be working as designed. A requirement has been opened and accepted to be resolved in a future release.

Working with Kerberos principals in RACF

The following RACF commands will allow you to know what Kerberos principals currently exist and to what RACF userids they are associated to.

Kerberos-based peer trust

We used *z/OS Security Server RACF Command Language Reference* for these commands.

SEARCH CLASS(KERBLINK)

The SEARCH command provides a nice consolidated list of the existing principals. All of the Kerberos principals on the system are listed.

RLIST KERBLINK * NORACF

The RLIST KERBLINK command shows the principals as well but not as nicely condensed.

RLIST KERBLINK *

If the NORACF parameter is not specified, the RACF info will be displayed. In the RACF info the APPLDATA field is displayed. The APPLDATA field lists the RACF id that the principal is associated to. Knowing what id a principal is associated to is valuable when needing to update the principal, such as changing the password. Any changes to the principal are made against the id the principal is associated to.

RLIST KERBLINK <principal-name>

The <principal-name> command will show the principal specified and the APPLDATA field. However, the command is not case sensitive. It will only work for principals that are ALL upper case. If any character in the principal name is lower case the command will return the following.

<principal-name> NOT FOUND

The command will be enabled for mixed case principals in a future release of z/OS.

Chapter 15. 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.

This chapter discusses the following topics:

- “Using WebSphere MQ shared queues and coupling facility structures”
- “Implementing WebSphere MQ shared channels in a distributed-queuing management environment” on page 222
- “Using WebSphere Business Integration Message Broker” on page 226

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 currently run WebSphere MQ for z/OS Version 5.3.1. We originally discussed our implementation of shared queues in our December 2002 edition. We continue that discussion by focusing on the usage and behavior of the coupling facility structures that support shared queues.

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, 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 six 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 six-member production queue sharing group. All of the queue managers in the group run WebSphere MQ for z/OS Version 5.3.1.

Our coupling facility structure configuration

We defined our MQ coupling facility structures to use two coupling facilities (CF2 and CF3) as defined in the prelist in the structure definitions. (See “Coupling facility details” on page 9 for details about our coupling facilities.)

The following is the structure definition for our CSQ_ADMIN structure:

```
STRUCTURE NAME(MQGPCSQ_ADMIN)
          INITSIZE(18668)
          MINSIZE(15000)
          DUPLEX(ENABLED)
          SIZE(18668)
          ALLOWAUTOALT(YES)
          PREFLIST(CF3,CF2)
          REBUILDPERCENT(1)
          FULLTHRESHOLD(85)
```

We also have the following four 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 MQ Integrator retail application

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)
          REBUILDPERCENT(1)
          FULLTHRESHOLD(85)
```

The other three 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, and MSGQ1 structures are recoverable and are backed up daily.

Testing the recovery behavior of the queue managers and 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 prelist, 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.

MQ structure recovery: During our normal coupling facility testing, the MQ CICS structure went into a failed state for valid reasons. This afforded us the opportunity to test MQ structure recovery. We issued the RECOVER CFSTRUCT command and the structure recovered with no errors.

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.

Suggested MQ maintenance

During the course of our testing, we applied the following WebSphere MQ APARs:

Table 15. WebSphere MQ APARs.

PQ72242	PQ77396	PQ96480
PQ72242	PQ77558	PQ99307
PQ72755	PQ78586	PQ98108
PQ74895	PQ82073	PQ98873
PQ75276	PQ82506	
PQ76590	PQ88137	
PQ77039	PQ94668	

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 00C51nnn. 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:

WebSphere Business Integration

- 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 a message is:

```
CSQ5003A !MQJA0 CSQ5CONN Connection to DB2 using DB1G pending, no active DB2
```

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 DBD1
```

- MQ abend reason codes that indicate a DB2 failure start with 00F5nnnn.

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 from 10000 to 15000 to accommodate the increase in usage.

Implementing WebSphere MQ 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. For z/OS V1R5, 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 192.168.49.31 192.168.49.32 192.168.49.33 192.168.49.34 92.168.49.36 192.168.49.38
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, which maps to that 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(DQMLNXP.TO.DQMMQGP) +
  REPLACE +
  CHLTYPE(SDR) +
  XMITQ(DQMMQGP.XMIT.QUEUE) +
  TRPTYPE(TCP) +
  DISCINT(15) +
  CONNAME('MQGP(1415)')

DEFINE CHANNEL(DQMLNXP.TO.DQMMQGP) +
  REPLACE +
  CHLTYPE(RCVR) +
  QSGDISP(GROUP) +
  TRPTYPE(TCP)
```

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(DQMLNXP.XMIT.QUEUE) +
  REPLACE +
  QSGDISP(SHARED) +
  CFSTRUCT(MSGQ1)
  TRIGGER +
  TRIGDATA(DQMMQGP.TO.DQMLNXP) +
  INITQ(SYSTEM.CHANNEL.INITQ) +
  USAGE(XMITQ) +
  STGCLASS(DQMSTG)
```

Example: The following are our definitions for the local sender channel and the remote receiver channel:

```
DEFINE CHANNEL(DQMMQGP.TO.DQMLNXP) +
  REPLACE +
  CHLTYPE(SDR) +
  XMITQ(DQMLNXP.XMIT.QUEUE) +
  QSGDISP(GROUP) +
  TRPTYPE(TCP) +
  DISCINT(15) +
  CONNAME(remote_client_host_name)

DEFINE CHANNEL(DQMMQGP.TO.DQMLNXP) +
  REPLACE +
  CHLTYPE(RCVR) +
  TRPTYPE(TCP)
```

Testing shared channel recovery

Based on the information in the MQ intercommunication guide, as well as information in the IBM Redbook, *WebSphere MQ in a z/OS Parallel Sysplex Environment*, SG24-6864, we tested several scenarios for shared channel recovery. For each scenario, we varied the DISCINT parameter of the channels in order to strike a balance between manual channel status observation and load-balanced channel starts. For our particular workload and environment, we set it to 60 seconds.

By observing the WLM goals for the Sysplex Distributor (using the NETSTAT VDPT command), we were able to ascertain the queue manager on which the inbound channel likely would start. In all of our tests, our sysplex workload mix caused queue manager CSQC on system JC0 to be favored as the destination of the Sysplex Distributor.

The following are the shared channel recovery scenarios that we tested, along with our experiences and observations:

Testing channel initiator failure

Action: Cancel the CHINIT address space.

Expected Results: The channel initiator fails, but the associated queue manager remains active. The queue manager monitors the failure and initiates recovery processing.

Actual Results: From a NETSTAT VDPT display, we observed that system JC0 had a WLM goal of 13 and JB0 had a goal of 12. All other queue manager systems had lower WLM goals. Thus, we expected the channel to start on JC0 (queue manager CSQC) and recover to JB0 (queue manager CSQB) when CSQCCHIN was cancelled.

With our DQM workload running over a shared inbound channel from our remote Linux host (queue manager LNXF) to the CSQC member of our MQGP queue sharing group, we canceled CSQCCHIN. The application continued to run successfully after the channel restarted on CSQB (as it had the highest WLM goal in the queue sharing group). After we restarted CSQCCHIN, when the channel timed out on CSQB, the next set of messages caused the channel to restart on CSQC.

The following messages appeared on the queue manager where CSQCCHIN was canceled:

```
CSQ3201E !MQJC0 ABNORMAL EOT IN PROGRESS FOR USER=
          CONNECTION-ID=CSQCCHIN THREAD-XREF=
CSQM052I !MQJC0 CSQMPCRT Shared channel recovery
          completed for CSQC, 1 channels found, 0 FIXSHARED, 1 recovered
```

Testing queue manager failure

Action: Cancel the MSTR address space.

Expected Results: The queue manager fails (failing the associated channel initiator). Other queue managers in the queue sharing group monitor the event and initiate peer recovery.

Actual Results: With our DQM workload running from our remote Linux host (queue manager LNXF) to the CSQC member of our MQGP queue sharing group, we canceled CSQCMSTR. The channel restarted on queue manager CSQB and the application continued to run successfully. After CSQCMSTR had completely restarted and channel DQMLNXP.TO.DQMMQGP timed out to CSQB, the next set of messages caused the channel to restart on CSQC (CSQCCHIN was restarted when CSQCMSTR restarted).

Testing DB2 failure

Action: Cancel the DB2 subsystem.

Expected Results: Channel state information is stored in DB2, so a loss of connectivity to DB2 becomes a failure when a channel state change occurs. Running channels can continue running without access to these resources. On a failed access to DB2, the channel enters the retry state.

Actual Results:

During normal operations, we lost the connection to the DB2 subsystem from queue manager CSQA (which is also a member of our MQGP queue sharing group). Subsequent attempts to display, start, or stop shared channels failed with the following error message:

```
CSQM294I - CSQA CSQMDRTS CANNOT GET INFORMATION FROM DB2
```

We then had to wait until the connection to DB2 was re-established in order to change the state of any shared channels. This corresponds to results discussed in *WebSphere MQ in a z/OS Parallel Sysplex Environment*.

Using WebSphere Business Integration Message Broker

WebSphere Business Integration Message Broker is the latest version of the product formerly known as WebSphere MQ Integrator. This section continues the discussion of our experiences with WebSphere MQ Integrator V2.1 from our December 2003 edition and includes our experiences migrating to WebSphere Business Integration Message Broker V5.0.

Note: To simplify the discussion, we'll refer to WebSphere Business Integration Message Broker as WBIMB, and WebSphere MQ Integrator as WMQI. However, these abbreviations are not officially sanctioned by IBM, so you should not use them to try to locate information or for product ordering, for instance.

Testing WMQI V2.1 on DB2 V8

We brought up one of our WMQI V2.1 brokers (CSQ1BRK) on DB2 V8 just to see if this would work. To do this, we first deleted the broker and its broker database on DB2 V7, and then recreated the broker using DB2 V8 for the broker and application databases. We also tested the scenario where the broker database is on DB2 V8 and the application databases is on DB2 V7 and had no problems. We tested using a variant of the Retail WMQI application that we described in our December 2003 edition.

Setting the `_BPXK_MDUMP` environment variable to write broker core dumps to MVS data sets

By default, broker core dumps are written to the home directory of the owner of the broker's started task in z/OS UNIX, with each dump in a separate file with a unique identifier. For example, in our sysplex, the started task owner for all brokers is MQSTEST, so the dumps are written to MQSTEST's home directory in the z/OS UNIX file system (/u/mqstest).

The problem is that these dumps are often quite large and can quickly fill up the HFS if it is not carefully monitored. Also, since we have four brokers running on different systems in the sysplex, with each broker using MQSTEST as the started task owner, it can often be difficult to determine which broker created which core dump.

We solved these problems by customizing our brokers to write core dumps to MVS data sets instead of to the started task owner's home directory. We specified a different data set for each broker's core dumps. To do this, we added the environment variable `_BPXK_MDUMP` to the ENVFILE file and also to the mqsicompif file (both of which reside in the broker's component directory in the

z/OS UNIX file system) so that if the broker is re-customized in the future, this change will not be lost. It is important to remember that the broker may have to be restarted to pick up changes to the ENVFILE.

We did the following to define the `_BPXK_MDUMP` environment variable and prepare the target MVS data sets to receive the broker core dumps:

1. We defined the `_BPXK_MDUMP` environment variable in the ENVFILE file to specify the name of the MVS data set to receive broker core dumps.

Example: We added the following line to the end of the ENVFILE file on system Z2 to cause broker core dumps on that system to be written to the MVS data set `WMQI.COREDUMP.Z2`:

```
_BPXK_MDUMP=WMQI.COREDUMP.Z2
```

2. Similarly, we also defined the `_BPXK_MDUMP` environment variable in the `mqsicompcif` file, between the `(ENVIRONMENTBEGIN)` and `(ENVIRONMENTEND)` tags.

Example: We added the following to the `mqsicompcif` file on system Z2:

```
(ENVIRONMENTBEGIN)
_BPXK_MDUMP=WMQI.COREDUMP.Z2
(ENVIRONMENTEND)
```

3. We allocated an empty MVS sequential data set for each data set name that we specified by a `_BPXK_MDUMP` environment variable. The data sets must already exist in order for core dumps to be written to them.

Example: The following is an example of the attributes we specified to allocate the MVS data sets:

```
Data Set Name . . . . : WMQI.COREDUMP.Z2

General Data
Management class . . : NOMIG
Storage class . . . . : STANDARD
Volume serial . . . . : PPRD0B
Device type . . . . . : 3390
Data class . . . . . : **None**
Organization . . . . : PS
Record format . . . . : FBS
Record length . . . . : 4160
Block size . . . . . : 4160
1st extent cylinders: 750
Secondary cylinders : 250
Data set name type :

Current Allocation
Allocated cylinders : 750
Allocated extents . : 1

Current Utilization
Used cylinders . . . : 0
Used extents . . . . : 0

SMS Compressible . . : NO

Creation date . . . . : yyyy/mm/dd
Expiration date . . . : ***None***
```

Note: We found that each time a broker writes a core dump, it simply appends to the end of the dump data set; therefore, it is important to monitor and clear out these data sets on a regular basis.

Resolving a EC6–FF01 abend in the broker

We ran into a problem where, immediately upon starting the broker, it would fail with an EC6–FF01 abend. We traced the cause of this problem back to the HFS for the broker component directory being completely full. Once we allocated more space for the HFS, the broker was able to successfully start.

Migrating WebSphere MQ Integrator V2.1 to WebSphere Business Integration Message Broker V5.0

We used the WBIMB V5.0 documentation in the online WebSphere Business Integration Information Center (publib.boulder.ibm.com/infocenter/wbihelp/index.jsp) and in the IBM Redbook, *Migration to WebSphere Business Integration Message Broker V5*, SG24-6995, to perform our migration.

Migration activities on the Windows platform

We performed the following activities to migrate to WBIMB V5.0 on the Windows platform:

1. We installed DB2 V8 on our Windows XP system, which replaced DB2 V7 that was already installed.

-
2. We had to migrate the existing broker and configuration manager databases so that they could be used with DB2 V8.

Example: We issued the following commands from the DB2 command line processor to migrate the broker databases from DB2 V7 to DB2 V8:

```
migrate database MQSIBKDB
migrate database MQSICMDB
migrate database MQSIMRDB
```

-
3. We performed some testing to verify that the WMQI V2.1 broker, configuration manager, and Control Center were still working following the database migration.

-
4. We installed the WBIMB V5.0 tool kit on our Windows XP system.

To do this, we followed the instructions in the online Information Center and the Redbook cited earlier. We strongly recommend that you read the relevant sections in the Redbook before starting your migration. As the instructions state, it is important to first uninstall the WMQI V2.1 Control Center *excluding data*, so that the data remains on the Windows system.

Migration activities on the z/OS platform

We have not yet tested the documented migration path for taking existing WMQI V2.1 brokers to WBIMB V5.0. Since the broker is a runtime component, we felt that simply deleting the V2.1 broker, recreating it as a V5.0 broker, and deploying the appropriate flows would be a satisfactory migration path. However, we do intend to try migrating one of our brokers by using the jobs and documentation that WBIMB V5.0 provides and we will report on any relevant experiences in a future test report.

For now, we deleted our WMQI V2.1 brokers and recreated them as WBIMB V5.0 brokers with Fix Pack 01. We noticed that the installation directory in WBIMB V5.0 is different than it was in WMQI V2.1 (for us, it was `/wbimb50/mqsi/V5ROM0` in V5.0, and `/wbimb50` in V2.1). Later on, after applying Fix Pack 02 and Fix Pack 03

(we did one right after the other), we noticed that the installation directory had again changed (now it was /wbimb50/V5R0M1), so we again had to update our jobs with the correct directory path.

As the documentation points out, it is important to note that Java version 1.4.0 is the minimum level that is required for WBIMB V5.1 on z/OS. See the section, “Checking the level of Java (z/OS),” in the online Information Center.

Applying WBIMB V5.0 Fix Pack 02 and Fix Pack 03

It is important to realize that FixPack 02 is actually a new release of WBIMB—namely, WBIMB 5.0.1. One difference that we found is the change in the installation directory path, as we mentioned earlier. This means that the JCL in the jobs generated by the mqsicustomize step needs to be updated to point to the new directory path. We found that the easiest way to do this was by recreating the broker and customizing it with a mqsicompCIF file that uses the correct installation library.

After applying the Fix Packs, we also ran into a problem where we could run the customization verification program (job name BIP\$JCVP) with no errors, but when we started the broker, it would come down after a minute or two with a 4039 user abend. We found that we had not listed all of the necessary directories in the LIBPATH in the mqsicompCIF and ENVFILE files (both of these files reside in the broker’s component directory in the z/OS UNIX file system). We corrected the problem and the broker started and ran successfully.

Some useful WBIMB Web sites

We found the following Web sites useful when working with WebSphere Business Integration Message Broker:

- README files for WebSphere MQ family products:
www.ibm.com/software/integration/mqfamily/support/readme/
- Fix Packs for WebSphere Business Integration Brokers:
www.ibm.com/software/integration/mqfamily/support/summary/wbib.html
- SupportPacs for WebSphere MQ family products:
www.ibm.com/software/integration/support/supportpacs/product.html

Note: We found SupportPac IP13 for WebSphere Business Integration Brokers to be particularly useful.

- IBM Redbooks: www.ibm.com/redbooks/

Chapter 16. Using IBM WebSphere Application Server for z/OS

This chapter describes our experiences using IBM WebSphere Application Server for z/OS and related products. Our test environment is now fully migrated to WebSphere Application Server for z/OS V5.1 running on z/OS V1R6. See “Migrating to WebSphere for z/OS V5.X” in our previous test report for information on our migration.

Note: References to WebSphere Application Server for z/OS V5.x appear in the text as “WebSphere for z/OS V5.x” or simply “V5.x.”

About our z/OS V1R6 test environment running WebSphere Application Server

Over the past few months, we have made a number of changes and updates to our WebSphere Application Server for z/OS test environment. In this chapter, we 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 V1R6 WebSphere test environment

This section provides an overview of our z/OS V1R6 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.

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 V1R6, our WebSphere test environment includes the products listed here:

- WebSphere Application Server for z/OS Version 5.1, service level W510211
- IBM SDK for z/OS, Java 2 Technology Edition V1.4.2 (February 09, 2005 Build Data, PTF UK00802)
- WebSphere Studio Workload Simulator V1.0
- WebSphere MQ for z/OS V5.3.1
- WebSphere Business Integration Message Broker for z/OS V5.0
- DB2 V7.1 with JDBC (PQ84404) and V8.1 with JDBC (PQ90211)
- CICS TS 2.3
 - CICS Transaction Gateway (CICS TG) V5.1
- IMS V8
 - IMS Connector V2.2

Software products on the workstation platform: On our workstations, we use the tools in this list to develop and test our Web applications:

- WebSphere Studio Application Developer, IE V5.1
- 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:

- Expanded our second test cell (T2) to include two additional systems (JB0 and JH0) (three total)
- Completed migration of all cells to WebSphere Application Server for z/OS V5.1
- Added another cell (QP) that spans 3 production nodes (J90, JC0 and JG0) that runs our MQ team's applications from were migrated from WebSphere Application Server for z/OS V4.0.1 plug-in
- Added a second HTTP Server configured with WebSphere Application Server for z/OS V5.1 plug-in to handle only SSL traffic
- Fully incorporated our use of Sysplex Distributor, HTTP Server with WebSphere Application Server for z/OS plug-in, and WebSphere Application Server for z/OS J2EE Servers across multiple nodes.

Our test and production configurations: In our environment, we have fully migrated to WebSphere for z/OS V5.1. Our current V5.1 setup contains four cells: T1 and T2 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.

Our T1 cell is configured as follows:

- Resides entirely on one of our test systems (Z1)
- Contains six different J2EE servers, each running different applications (as described below)

Our T2 cell is configured as follows:

- Generally resides on one of our test systems (Z2), but also has nodes configured on two additional systems (JB0 and JH0)
- Contains six different J2EE servers, each running different applications (as described below)

Our P1 cell is configured as follows:

- Spans four production systems in our sysplex (J80, JB0, JF0, and JH0)
- Contains six different clusters, each of which spans all four 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 Trade3 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

| Figure 47 on page 234 shows the server address spaces in our P1 cell.

| **Note:** The wsp1s1 cluster is not shown in the diagram.
|

WebSphere Application Server for z/OS

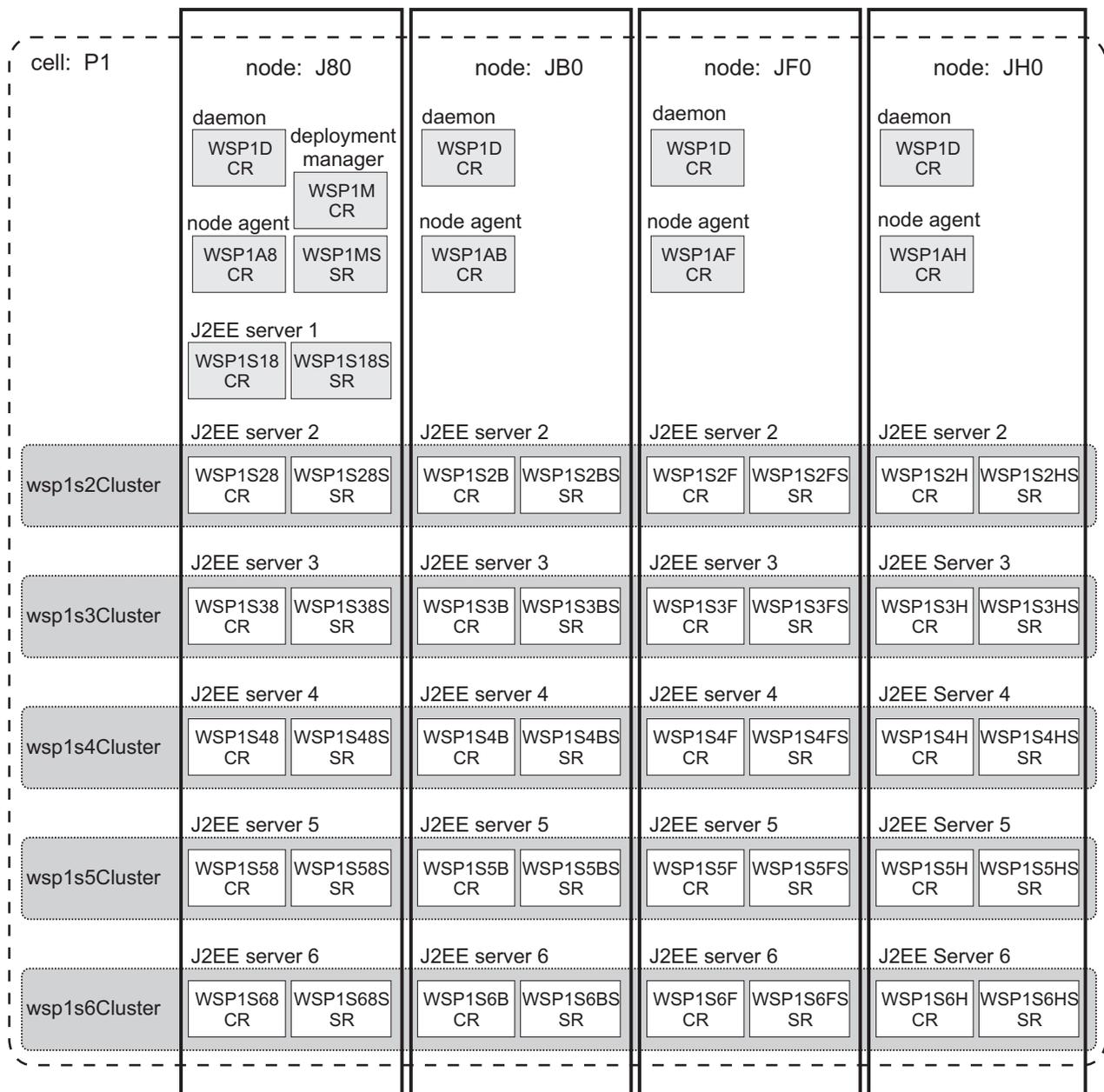


Figure 47. Our WebSphere for z/OS V5.1 configuration

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
 - Sn** J2EE server control region, instance *n*
- y* System identifier:
- 1** Z1 (test)
 - 2** Z2 (test)
 - 8** J80 (production)
 - B** JB0 (production)
 - F** JF0 (production)
 - H** JH0 (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 WebSphere production cell 1 J2EE server server region 1 on system J80.

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 describe other changes and updates to our WebSphere test environment.

Migrating WebSphere Application Server for z/OS JDBC from DB2 V7 to DB2 V8

We migrated our WebSphere Application Server for z/OS servers from using DB2 7.1 to DB2 8.1 for JDBC and experienced no problems with their usage. The items we changed are described below. See the IBM WebSphere Application Server Version 5.X Information Center, available at publib.boulder.ibm.com/infocenter/wasinfo/index.jsp for a complete step-by-step guide to assist you in the proper setup for DB2 JDBC. Also see “Using DB2 UDB JCC Connectors” for configuring UDB connectors.

We did the following to migrate to using DB2 V8:

- Created a new `db2sqljdbc.properties` file for DB2 V8
- Updated the following environment variables: `DB2390_JDBC_DRIVER_PATH`, `DB2UNIVERSAL_JDBC_DRIVER_PATH`, and `DB2UNIVERSAL_JDBC_DRIVER_NATIVEPATH` to point to the appropriate DB2 V8 JDBC/JCC directories and `DB2SQLJPROPERTIES` to point to the new `db2sqljdbc.properties` file
- Changed all DB2 plan references to reference our DB2 V8 jdbc plan name. This needs to be done in the new `db2sqljdbc.properties` file as well as the Custom Properties for the various JDBC data sources defined in WebSphere Application Server for z/OS.

Using DB2 UDB JCC Connectors

Support for DB2 Universal JDBC Type-2 and Type-4 drivers is now available. We have taken advantage of the new connectors for z/OS and experienced no problems with their usage. To use these WebSphere Application Server for z/OS

WebSphere Application Server for z/OS

must be at service level W502004 or higher and APAR PQ80841 for DB2 V7 for z/OS applied. Support is provided with DB2 UDB V8 for z/OS.

See "Enabling WebSphere Application Server V5 for z/OS to use the DB2 Universal JDBC Driver", TD101663, for a complete step-by-step guide to assist you in the proper setup for using the Type-2 and Type-4 connectors with WebSphere Application Server V5. This white paper is available on the IBM Techdocs Web site at: <http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/TD101663>

See also the Redbook: "DB2 for z/OS and WebSphere: The Perfect Couple", SG24-6319 for full details on using these connectors in the WebSphere environment. This redbook is available at www.redbooks.ibm.com.

Migrating to CICS Transaction Gateway Connector V5.1

We have migrated our CICS Transaction Gateway (CTG) from Version 5.0.1 to Version 5.1. Our migration was a very simple and straight forward process, and we experienced no problems with the current level. One particular note is that the CTG 5.1 Daemon will not run with JDK 1.4.2. You must use JDK 1.3.1 or 1.4.1 for this process. The CTG J2EE Connector runs well in WebSphere Application Server 5.1 when you use JDK 1.4.2 there.

Installing CTG 5.1

Our installation of CTG 5.1 is almost the same as with previous levels of CTG. The z/OS Administration book, available at <http://www.ibm.com/software/ts/cics/library/>, covers this process very well with step-by-step details. We made the same updates to the ctgenvar file, based upon updates we had done for CTG 5.0.1.

Updates to CTG Daemon startup procs

CTG 5.x now handles the output messages better. By default, client connect and disconnect messages are not printed.

We run our CTG Daemon as a Started Task using JCL and direct output to HFS files. In the past, we have had to run with STDERR set to /dev/null to keep the HFS from filling up with these messages.

With CTG V5.1, we now enable STDERR to an HFS file, allowing us to see any important error messages we might have missed in the past.

Installing J2EE CICS ECI Connector in WebSphere Application Server V 5.1

By way of the instructions in the CTG V5.1 Administration Guide book, uninstall any previous version of the CICS Connector prior to installing this version.

References

Documentation for CTG 5.1 is available from <http://www.ibm.com/software/htp/cics/ctg/library/> including the CICS Transaction Gateway - z/OS Administration - Version 5.1, SC34-6191-01, book.

This site also includes links to various other documents on CICS Transaction Gateway, including White Papers, Redbooks, and Configuration Guides.

Enabling Global Security and SSL on WebSphere Application Server for z/OS

We have enabled Global Security and Secure Sockets Layer (SSL) on each of our WebSphere Application Server for z/OS 5.x Cells. Security is extensively integrated

into both the z/OS operating system and WebSphere Application Server for z/OS. There is a considerable amount of setup work required to get WebSphere Application Server for z/OS installed and initially running on z/OS. While that alone can be daunting enough, even more planning and setup work is required before enabling Global Security. Proper planning is essential.

A number of issues relate to WebSphere Application Server for z/OS security, and it is highly recommended that you pick up the latest WebSphere Application Server for z/OS service level.

In our WebSphere Application Server 5.1 environment, we are using security in the following areas:

- LocalOS (local operating system) for our User Registry, with RACF for SAF
- Unique userids for each WebSphere Application Server, including Daemon, Deployment Manager and Node Agents
- Server Certificates managed in RACF key rings
- SSL Transport Handlers enabled for all J2EE servers
- SSL enabled through HTTP Server with WebSphere Application Server for z/OS plug-in
- EJBROLE authorization checking
- Component and container managed authorization as appropriate for DB2, CICS and IMS resources.

Global Security — "the Big Switch"

Global Security can be considered a "big switch" for WebSphere Application Server for z/OS. Disabled, many security checks are bypassed. When "switched on", things can appear that you might not have thought of or planned for.

For example, with global security disabled, the web-based Admin Console application will prompt for and accept virtually anything for a userid (even a userid that is not valid on the system) and no password. When global security is enabled, you will need a valid userid/password to enter. If EJBROLE authorization is enabled, further setup is also required to authorize the userid. It's possible to "lock yourself out" of the Admin Console application. (Don't worry, WebSphere Application Server for z/OS provides some ways to get in the back-door and disable global security. We found out that this function also works well! If this should happen to you, search the WebSphere Application Server InfoCenter for "securityoff" for full details on how to perform this task.

We highly recommend you test your security setups thoroughly before implementing them in a production environment. After enabling global security, we also recommend a close review of the output from all of the WebSphere Application Server for z/OS servers. Enabling global security also enables secure communications for administrative functions between the various WebSphere Application Server for z/OS address spaces. This is predominant in a Network Deployment setup that uses separate servers for the Deployment Manager and Node Agents.

Migrating from WebSphere Application Server for z/OS 5.0.2 to 5.1 with Global Security enabled

We didn't have too many problems enabling global security on our WebSphere Application Server for z/OS test cell that was originally installed using WebSphere Application Server for z/OS 5.1. We also didn't have any problems when we migrated our production cell from WebSphere Application Server for z/OS 5.0.2 to

WebSphere Application Server for z/OS

5.1 **without** global security enabled. However, we had a number of problems during migration of another WebSphere Application Server for z/OS test cell from WebSphere Application Server for z/OS 5.0.2 to 5.1 **with** global security enabled.

We're happy to report that these issues have now been resolved. We were able to successfully migrate our test cell with global security enabled from WebSphere Application Server for z/OS 5.0.2 to 5.1 with the following:

- WebSphere Application Server for z/OS 5.0.2 W502017 service level
- WebSphere Application Server for z/OS 5.1 W510205 service level
- Test fix for APAR PK04797
- Formal fix now available in PTF UK03577, which is associated with service level W510212.

References

See the IBM WebSphere Application Server Version 5.1 Information Center, available at publib.boulder.ibm.com/infocenter/ws51help/index.jsp for full instructions and details on the various aspects of security with WebSphere Application Server Version 5.1.

Also see the following IBM Techdocs at www.ibm.com/support/techdocs/

- "Enabling Global Security in WebSphere Application Server V5 for OS/390 and z/OS" found at <http://www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/TD101150>
- "WebSphere for z/OS Security Overview" found at <http://www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/PRS1054>
- "RACF Tips for customizing WebSphere for z/OS Version 5.0.2" found at <http://www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/TD101118>.

Using the WebSphere Application Server for z/OS 5.x plug-in for HTTP Server and Sysplex Distributor with our WebSphere Application Server for z/OS J2EE Servers

As part of our configuration for WebSphere Application Server for z/OS 5.x, we have incorporated the Sysplex Distributor and HTTP Server with WebSphere Application Server for z/OS plug-in to "front end" our WebSphere Application Server for z/OS J2EE Servers across multiple systems.

Using the Sysplex Distributor and HTTP Server with WebSphere Application Server for z/OS plug-in provides us with a number of advantages, including:

- Scalability
- High availability
- Workload balancing
- Single point of input for many J2EE applications
- Session affinity routing
- Static content serving and caching.

For full information on using and configuring the WebSphere Application Server 5.x Plug-in and the Sysplex Distributor, please see the following:

- WebSphere Application Server Library: <http://www-306.ibm.com/software/webservers/appserv/was/library/>
- WebSphere Application Server 5.x InfoCenter at: <http://publib.boulder.ibm.com/infocenter/ws51help/index.jsp>

- WSC TechDoc PRS829, "Configuring and Troubleshooting the WebSphere for z/OS Version 5 HTTP Server plug-in" available at: <http://www-03.ibm.com/support/techdocs/atmastr.nsf/WebIndex/PRS829>
- IBM HTTP Server V5.3 for z/OS Library at: <http://www-306.ibm.com/software/webservers/httpservers/doc53.html>

Using the HTTP Server with WebSphere Application Server for z/OS plug-in along with our J2EE Servers

There are numerous ways of configuring WebSphere Application Server for z/OS J2EE servers and applications. Many of these are done for reasons such as scalability, application and/or resource isolation.

We use a combination of techniques for our setups, including

- Multiple J2EE servers per system for application and resource isolation
- Multiple Server Regions per J2EE server for scalability and fail-over
- J2EE Server clusters spanning multiple systems for scalability and system fail-overs.

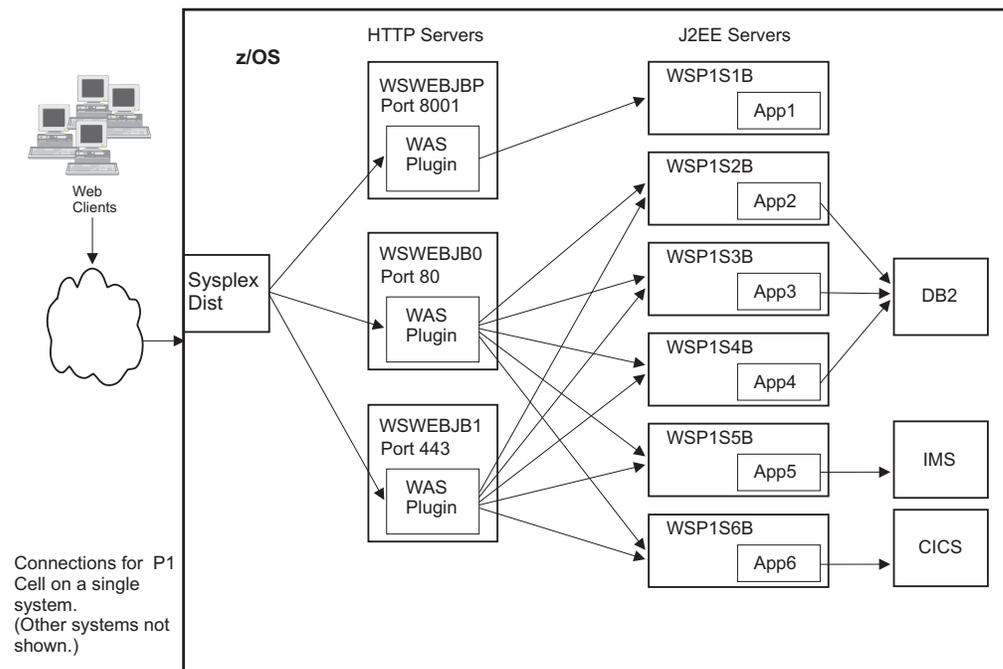


Figure 48. Servers in one system of our WebSphere Application Server for z/OS P1 Cell (production)

The above diagram depicts the various servers in one system of our WebSphere Application Server for z/OS P1 Cell (production).

- The Sysplex Distributor "front ends" all incoming requests
- Sysplex Distributor sends the request to one of the HTTP Servers, each listening on a different port
- The HTTP Server with WebSphere Application Server for z/OS plug-in sends the request to one of the J2EE Servers.
 - WSWEBJBP is configured to handle only requests for App1
 - WSWEBJB0 is configured to handle only HTTP (non-SSL) requests for App2, App3, App4, App5, and App6

WebSphere Application Server for z/OS

- WSWEJB1 is configured to handle only HTTPS (SSL) requests for App2, App3, App4, App5, and App6.
- Multiple J2EE Servers are used to provide:
 - Application isolation
 - Resource isolation
 - Increased client capacity.
- All J2EE Servers are configured as part of a WebSphere Application Server for z/OS J2EE Server cluster. Each cluster has one J2EE server on each system.

Bottlenecks in our HTTP Server

When initially using the HTTP Server with WebSphere Application Server for z/OS plug-in to "front end" all of this activity, bottlenecks appeared in the HTTP Server.

To help resolve this, we needed to understand:

- The differences between the threading mechanism in the two types of servers
- The amount of concurrent users expected through an HTTP Server

Client Handling differences between HTTP Server and WebSphere Application Server for z/OS J2EE Server on z/OS: The HTTP client handling mechanism is very different for the HTTP Server from the WebSphere Application Server for z/OS J2EE servers. The concurrent client capacity that a WebSphere Application Server for z/OS J2EE server can handle can be very much higher than the HTTP Server.

In a J2EE server, client requests are initially received by the J2EE server's control region. The J2EE server's control region then queues the request to the server region(s) through WLM. Once queued, the control region thread is available to handle another request. Multiple J2EE server regions can be configured for each J2EE control region, also increasing the client capacity.

The HTTP Server, on the other hand, uses a thread for the duration of each request. This includes the time spent processing the request after it has been forwarded to a J2EE server. The thread waits for the J2EE server to respond and returns this to the client. The HTTP Server's thread is then available to handle another request.

Concurrent client traffic through HTTP Server: We also needed to take a closer look at how much traffic an HTTP Server was expected to concurrently handle, including the following considerations:

- The number of J2EE servers that the plug-in will service.

A single HTTP Server can handle applications running in clustered J2EE servers across multiple nodes. The same HTTP Server can also handle multiple applications that are spread across multiple J2EE servers in a single node. When combined, this vertical and horizontal scaling can multiply the number of concurrent requests the J2EE servers can handle.
- The number of Server Regions, if configured for multiple per J2EE server regions.

All application work is processed in the J2EE server's server region. Running multiple server regions increases the amount of work that can be concurrently handled.
- The number of threads each J2EE server region is running.

Increasing the number of threads can have the same impact on the HTTP Server as running with multiple server regions.

- Failure of other components in the cluster and service updates.
If one of the systems goes down, either because of a failure or for service updates, work will be routed to the remaining server(s). Allow some headroom in each HTTP Server to handle this increase.

As an example of these considerations, let's say an HTTP Server is front ending one J2EE Server that is running with 4 Server Regions, each running 10 threads. At any one time, there is threading capacity for at least 40 clients running between these J2EE application servers.

If the HTTP Server is set at the default of 40 threads, it is at its limit of capacity for concurrent clients.

You can use the z/OS MODIFY command,

```
F server_job_name,appl,-d stats
```

to display statistics about the HTTP Server's thread usage.

Statistics can also be viewed using a web browser if the HTTP Server is configured for access to its Server Activity Monitor (Service /Usage* directive). If using this web access, it's recommended access to this function be protected and/or secured (for example, using Protect /Usage* directives).

See the HTTP Server Planning, Installing, and Using Version 5.3, available at <http://www.ibm.com/software/websphere/httpservers/library.html>, for full details.

Review the J2EE server's control region output for the following information:

```
BB000234I SERVANT PROCESS THREAD COUNT IS 9.    (this is determined by the
           server_region_workload_profile setting.
BBOM0001I      server_region_workload_profile: IOBOUND.
BBOM0001I      wlm_maximumSRCCount: 1.
BBOM0001I      wlm_minimumSRCCount: 1.
```

Scaling up the HTTP Server with WebSphere Application Server for z/OS plug-in along with our J2EE Servers

To help increase the overall throughput of our Web serving setups, we have implemented a few things:

- Increasing the MaxActiveThreads setting for the HTTP Server
- Splitting HTTP and HTTPS traffic between two HTTP Servers
- Splitting traffic between multiple HTTP Servers.

Increasing the MaxActiveThreads for the HTTP Server: One simple way to increase the concurrent client capacity of the HTTP Server is to increase the MaxActiveThreads setting in the HTTP Server's configuration file (httpd.conf). The default is 40 threads. It is not recommended to increase the value to greater than 150-200.

Splitting HTTP and HTTPS traffic between two HTTP Servers: The MaxActiveThreads setting for the HTTP Server sets the total number of threads the HTTP Server is running. This might not be the total number of clients that it can concurrently process. When the HTTP Server is configured for both HTTP and SSL connections, it splits the MaxActiveThreads count into two thread pools, one to handle requests for each type. If needed, the server will move a thread from one pool to the other, however, this balancing only occurs when the thread finishes processing a request. This generally works well when you have a relatively steady flow of both types of requests (HTTP and SSL). However, the HTTP Server can get

WebSphere Application Server for z/OS

"caught off-balance." For example, when the HTTP Server is first started with MaxActiveThreads set for 40, two pools are created with 20 threads each. If all the current requests to the server are for non-SSL connections, only 20 threads will be available for handling these requests. Threads in the SSL pool will not be potentially moved until there is an SSL request. If the server never receives an SSL request, there will never be any thread pool movement.

To help minimize this condition, we run two HTTP Servers for our "public" applications in our production setups,

- One configured to handle only non-SSL traffic
- One configured to handle only SSL traffic

The HTTP Server configured for port 80 was disabled for SSL by changing the "sslmode" directive to off in the HTTP Server's configuration file. All threads in this server are dedicated to running non-SSL connections and no pool balancing occurs.

We created a second HTTP Server that was configured with only SSL enabled. This was done by setting the "normalmode" directive in the HTTP Server's configuration file to "off". This server is set to listen only on port 443 for SSL requests. All threads in this server are dedicated to running SSL connections and no pool balancing occurs.

Multiple HTTP Servers for multiple applications: On each system within our WebSphere Application Server for z/OS cell, we run multiple J2EE Servers to handle the various applications. While most applications are considered "public" applications, we have some that are considered "internal" applications. The "public" applications are expected to be normally accessed using the default ports (80 for non-SSL, 443 for SSL). The "internal" applications can be accessed using non-default ports (such as 8001).

While a single HTTP Server with WebSphere Application Server for z/OS plug-in can be configured to access all applications in all J2EE servers, we have multiple HTTP Servers, each configured to handle only selected applications.

The HTTP Servers configured to handle the "public" applications are generally configured for ports 80 and/or 443. The HTTP Servers configured to handle the "internal" applications are configured to run on other ports (such as 8001).

This helps to isolate the expected traffic for each of the HTTP Servers.

TrustedProxy setting in J2EE Server's WebContainer

If using the WebSphere Application Server for z/OS plug-in to front end the J2EE servers, make sure to set the TrustedProxy custom property for the J2EE Server's transport handlers to "true". This setting enables the application server to use the private headers that the Web server plug-in adds to requests. We set this property as a web container's Custom Properties page so all transports will support private headers. Otherwise, it needs to be added as a custom property for each transport.

If you try to use private headers without adding the TrustedProxy property, they will be ignored. If the private headers are ignored, the application server might not locate the requested application.

HTTP Server SSL setup needs J2EE Server's CA

When using the HTTP Server with WebSphere Application Server for z/OS plug-in to front end the J2EE servers for SSL connections, make sure that the SSL setup

for the HTTP Server contains the CA (Certificate Authority) that signed the J2EE server's certificate. Otherwise, the SSL handshake between the plug-in and the J2EE Server will fail.

Customizing the WebSphere Application Server for z/OS plug-in configuration file (pluginfg.xml)

We made a number of changes to the configuration file that is used by the WebSphere Application Server for z/OS plug-in (pluginfg.xml).

We started by generating a pluginfg.xml file using the WebSphere Application Server for z/OS Admin Console application or by using the GenPluginCfg.sh script. This is also done after updates to the configuration that would affect the settings, such as adding/deleting applications, servers, ports, and others.

The pluginfg.xml file that is generated by WebSphere Application Server for z/OS is a good starting point as it contains default settings to accommodate all virtual hosts, applications, servers and clusters configured within the cell. The WSC TechDoc PRS829 document does a very good job of describing much of the customization of this file. Also see the information in the WebSphere Application Server for z/OS InfoCenter for descriptions of some of the additional settings that are not included in the generated file, such as <BackupServers>.

We customize this configuration file by making some or all of the following changes (see specifics for each below):

- Refresh interval changed for production systems
- Logging directives updated to use the HTTP Server's logging directory
- Settings are customized for only the ports, servers, clusters and applications we wish the particular HTTP Server to handle
- ConnectTimeout lowered.

Our WebSphere Application Server for z/OS plug-ins do not use the generated configuration files directly. Be aware that if you do, regeneration of the plug-in configuration file will overwrite any changes you have made. A copy of this file is created for each of our HTTP Server setups. For HTTP Servers that are being front ended by the Sysplex Distributor, all servers on a particular port use a common file.

Once the configuration file has been initially modified and placed in service, it may be more cumbersome to repeat the editing process for future changes; this depends on the amount of initial changes. We regenerated the configuration file using the WebSphere Application Server for z/OS Admin Console. Next, the updated file is compared with the previously generated file. Finally, changes are manually applied to the customized configuration files.

Setting the RefreshInterval: The "RefreshInterval" setting in the Config section determines the interval at which the plug-in will check for updates to the configuration file. The default for this is 60 seconds. We generally increase this value on our production systems to at least 300 (5 minutes) since the configuration rarely changes.

Logging settings: Logging directives are updated to use the HTTP Server's logging directory. This directory is located on a system-specific (non-shared) HFS for performance reasons. The date and PID of the HTTP Server is appended to the supplied name. This also helps correlate the log file with other logging files produced by the HTTP Server.

Configuring applications: In our setups, each HTTP Server with WebSphere Application Server for z/OS plug-in is configured to handle only certain applications. For each plug-in configuration, after starting with a copy of the generated configuration file, we delete the applications that we don't want the server to handle, leaving only the applications that we want the server to handle. The generated configuration file contains all applications, servers, and so on.

For example, we did not want our plug-in configured in HTTP Servers using Port 80 or 443 (SSL) to handle access to our "internal" applications. Since these applications run in separate J2EE servers from other applications, all we needed to do was remove the references to these applications and server clusters. In this way, the plug-in never "sees" the applications.

For this type of update, we removed the settings for the "internal" applications in the following areas:

- VirtualHost entries from the VirtualHostGroup for the J2EE servers running the applications
- ServerCluster entries for the J2EE servers running the applications
- URIGroup entry defined for the applications
- Route entry for the VirtualHost, ServerCluster and URIGroup.

Modifying the ConnectTimeout value: We generally lower the value(s) for the "ConnectTimeout." This is the time the plug-in will wait for a connection to the J2EE Server. The default in the generated plugin.cfg file is 60 seconds. If the connection to the J2EE server times out, the server will be marked as unavailable. After timing out a particular server, the plug-in will try to connect with the next server in the "PrimaryServers" list for the application. As we added more J2EE servers to the cluster, this time could become excessive, especially during initial startup of the HTTP Server. For instance, if one of the "PrimaryServers" is not available, the plug-in will hold a request for one minute waiting to connect to this server before attempting to send the request to an available server. Users aren't always that patient!

To minimize this impact, we generally change this value to 10 seconds.

The server will be marked as unavailable until the "RetryInterval" value is met. If all "PrimaryServers" have been marked as unavailable, the plug-in will then try to connect to a "BackupServer."

Improving Static Content performance

Unfortunately, the WebSphere Application Server for z/OS J2EE servers are not the best at handling static content. They really are geared for providing dynamic content. The inclusion of the HTTP Server into our WebSphere Application Server for z/OS setups added some better ways of handling the static content for our web applications. These include using the HTTP Server as an origin server for the static content via Pass directives and using the Fast Response Cache Accelerator (FRCA). The WebSphere Application Server for z/OS plug-in for the HTTP Server can also be configured to handle some caching using Edge Side Include (ESI).

To help minimize the performance impact of running requests through the HTTP Server, we have moved most of the handling of static content of our web applications over to the HTTP Server's FRCA. Using FRCA greatly improves the overall performance of our web applications. After the initial request for static content, it is placed in FRCA. Further requests for it does not propagate beyond the TCP/IP stack. This helps to use the "right tool for the right job".

Implementing this is not necessarily trivial, and takes some considerations and planning and might require intimate knowledge of the application. Ideally, separation of static content from dynamic content should be taken into account from the beginning of the design of the application. Static content can be easily separated from dynamic content, both logically (based on URLs) and physically. For many reasons, static content is generally included with the deployable J2EE EAR or WAR file.

See "Handling Static Content in WebSphere Application Server" available at: http://www.ibm.com/developerworks/websphere/techjournal/0211_brown/brown.html on the IBM Developer's Domain for more information on various techniques for separating Static from Dynamic Content in your applications.

Experiences with using HTTP Server and FRCA for Static Content handling:

As we continue to try to improve the overall performance of our web applications, we have tried and used a variety of techniques for caching. Here are some of our experiences with using the HTTP Server with WebSphere Application Server for z/OS plug-in and FRCA for handling static content.

- If static content is deployed separately from the J2EE application, make sure you have a process in place to keep the two in sync when one or the other is updated. If static content is extracted from the J2EE application after deployment, make sure to have a process in place to perform this after application updates.
- Don't forget about less common types of static content. HTML files, GIF and JPG images easily come to mind, but don't forget about others such as CSS and WAV files. Also, keep an eye out for the various file extensions that may exist in an application. For example, HTML files may have an extension of *.html or *.htm,
- Make sure that any directives added for static content (Pass, Map, and others) do not interfere with or are "hidden" by service directives used to send URLs for dynamic content to the WebSphere Application Server for z/OS plug-in. Placement within the HTTP Server's configuration file is also important.

For example, to allow for static content from an application to be handled by the HTTP Server, we added the following directive:

```
Pass /OurApp/images/* /ws/images/OurApp/*
```

The Service directive to send dynamic requests to the WebSphere Application Server for z/OS plug-in were:

```
Service /OurApp/* /ws/p1wassmpe/bin/ihs390WAS50Plugin_http.so:service_exit
```

Because the URL comparison string for the Service directive (/OurApp*) is more generic than the one for the Pass directive (/OurApp/images/*), the Pass directive needs to be placed ahead of the Service directive in the configuration file.

Otherwise, all requests for the application will match the Service directive first and be processed. The Pass directive will never be compared with, effectively "hidden".

- With our HTTP Server running on the same system as the J2EE server, it looked simple to have the HTTP Server directly access the deployed J2EE application's static content, rather than "extract" it. The idea was to use Pass and FRCA directives to point to the static content directly in the file system where the J2EE application was deployed, generally, //deployedApps/. While this can be a workable solution, we ran into a few glitches in the process. Some of the major issues were:
 - HFS file ownerships and permission bits
 - File translations / AddType directives.
- HFS file ownerships and permission bits.

WebSphere Application Server for z/OS

Be careful of HFS file ownerships and permission bits. When WebSphere Application Server for z/OS deploys an application, the files generally have the WebSphere Application Server for z/OS Administrator's uid/gid and permission bits of 640. The userid the request is being handled under in the HTTP Server (Userid directive) may not have access to these directories/files. Proper permissions must also be given for all higher level directories, which may lead to unintentional access to other WebSphere Application Server for z/OS configuration data and extreme care should be taken.

- File translations

Due to WAR and EAR packaging of J2EE applications and the WebSphere Application Server for z/OS deployment process, the files for a deployed application are generally stored in ASCII. The "SimpleFileServlet" provided in J2EE Web Containers will not perform any translation of the files, so the files will be returned in the correct code page. The HTTP Server uses AddType directives to determine whether to translate a file before returning it when operating as an origin server (using Pass directives). By default, AddType directives will translate files with *.html extensions from EBCDIC to ASCII before returning them to the requestor. (Content from a J2EE server back through the HTTP Server with WebSphere Application Server for z/OS plug-in is not translated).

If the HTML files are stored in ASCII on the z/OS system, there are a few choices:

- Translate all HTML pages using a tool such as "iconv"
- Change the AddType directive(s) in the HTTP Server's configuration file.

Changing the HTTP Server's AddType directive will affect all files of this type. If the HTTP Server is handling static content for multiple applications, each will be affected.

Sysplex Distributor usage with HTTP Server / WebSphere Application Server for z/OS J2EE Servers

We have configured our J2EE servers to run in clusters. Each cluster has one J2EE server configured on each of 4 systems in our sysplex for our production setup. Each system in the cluster also runs an HTTP Server configured with the WebSphere Application Server for z/OS plug-in to "front end" the J2EE Servers. We use the z/OS Sysplex Distributor to balance requests to the various HTTP Servers across the systems.

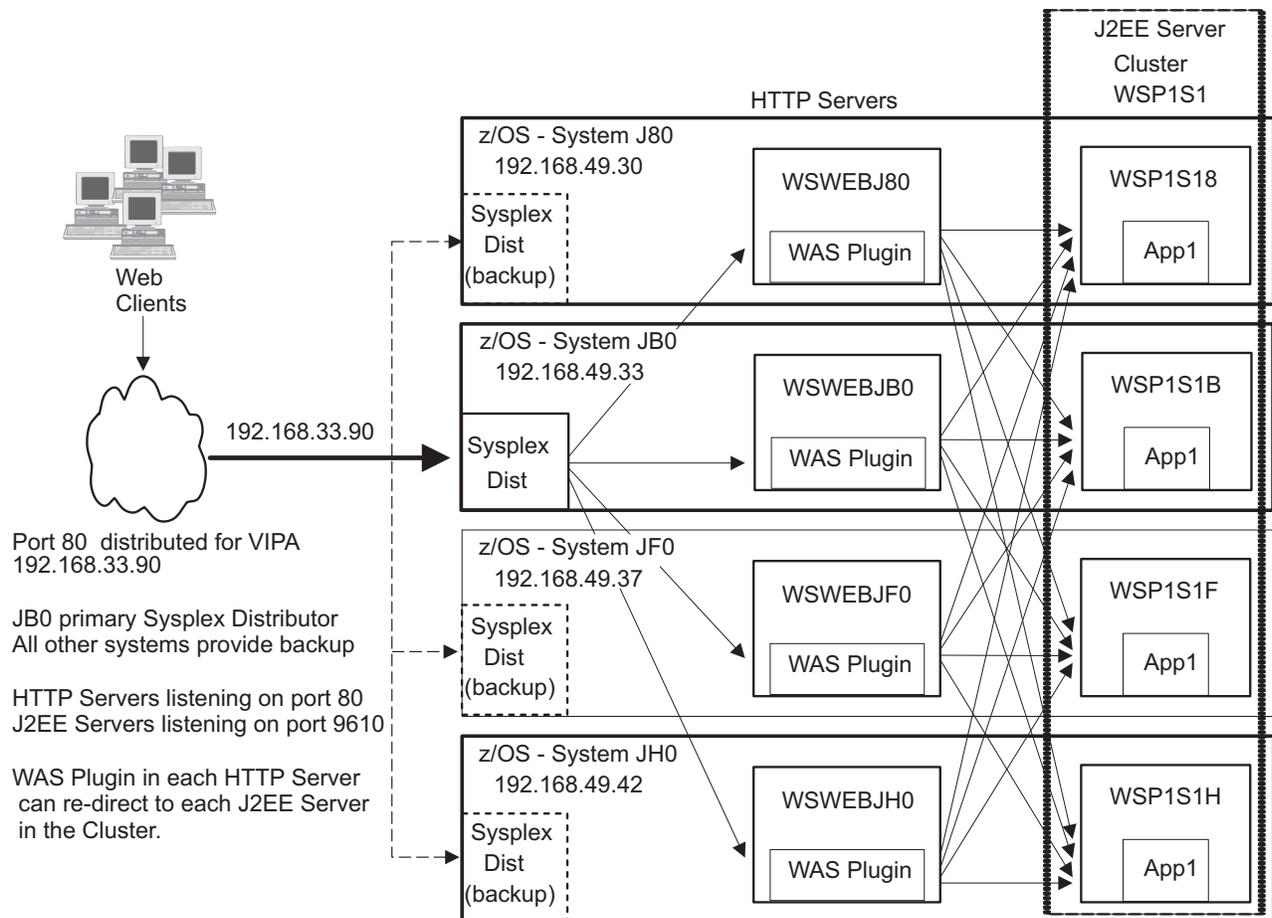


Figure 49. One J2EE Server Cluster (WSP1S1) in our WebSphere Application Server for z/OS P1 Cell (production)

Figure 49 shows one J2EE Server Cluster (WSP1S1) in our WebSphere Application Server for z/OS P1 Cell (production).

- The Sysplex Distributor "front ends" all incoming requests
- JB0 is the primary distributor. All other systems are configured to be backups
- Sysplex Distributor sends the request to one of the HTTP Servers
- The HTTP Server with WebSphere Application Server for z/OS plug-in sends the request to one of the J2EE Servers

On our production setups, we distribute various HTTP Server ports (80, 443, 8001, and so on). We have also configured some of our WebSphere Application Server for z/OS Administrative ports to be front ended by the Sysplex Distributor. These include the Admin Console application and the HTTP/HTTPS Transport Handler ports used by certain J2EE server clusters whose applications don't require session affinity.

One system is used as the "primary" Sysplex Distributor, with all other systems configured as "backups". The Sysplex Distributor function is automatically moved to another system in the event of a failure of the primary system (or one of the backups).

To monitor the Sysplex Distributor usage, use the netstat command. See the "z/OS Communications Server IP Administrator's Commands" at: <http://www.ibm.com/servers/eserver/zseries/zos/bkserv/> for details.

WebSphere Application Server for z/OS

See the following for details on planning, implementing and monitoring the Sysplex Distributor for z/OS:

- WSC TechDoc WP100312, "Use of WebSphere for z/OS with Sysplex Distributor" available at:
<http://www.ibm.com/support/techdocs/atmastr.nsf/Web/Techdocs>
- Various IBM Communications Server product publications, available at:
<http://www.ibm.com/servers/eserver/zseries/zos/bkserv/>

Configuring for session data persistence: We have configured our J2EE Servers that run applications using HTTP Sessions to use DB2 as a persistent store for the session data. The WebSphere Application Server for z/OS plug-in for the HTTP Server will generally reroute any requests with a "Session Affinity" back to the server that previously processed the request, since the session data is likely still in that server's memory. Backing the session data in DB2 helps allow for failure of a J2EE server. This also allows better control of memory consumption by a J2EE Server by limiting the amount of session data held in memory, without sacrificing long running sessions.

We maintain tables in DB2 just for the session data, separate from other application tables. Each J2EE server is also configured with a JDBC Datasource strictly for the connections to DB2 for the session management. This helps to minimize contention with the application for DB2 resources.

See the WebSphere Application Server Library at: [http://www-306.ibm.com/software/webservers/appserv/WebSphere Application Server for z/OS/library/](http://www-306.ibm.com/software/webservers/appserv/WebSphere%20Application%20Server%20for%20zOS/library/) for complete information on configuring session persistence.

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 and OS/390 documentation, available at http://www.ibm.com/software/webservers/appserv/zos_os390/library/
- IBM WebSphere Application Server Version 5.X Information Center, available at publib.boulder.ibm.com/infocenter/wasinfo/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/websphere/httpservers/library.html>
- IBM WebSphere Studio Workload Simulator documentation, available at www.ibm.com/software/awdtools/studioworkloadsimulator/library/

Specific documentation we used

Documentation to assist you with the usage of your product is available in many places. We have found that the Washington Systems Center documentation is very good and very often this same information is also in the information center. While we offer a set of generic links to documentation, see "Where to find more

information” on page 248 for more information, we also wanted to take this opportunity to highlight the specific documentation we used and found especially useful.

For our current WebSphere for z/OS V5.1 configuration, we found the following technical documents were especially good at getting us up and running quickly:

- IBM TechDocs are available at www.ibm.com/support/techdocs/. We used the following specific TechDocs:

- “Enabling Global Security in WebSphere Application Server V5 for OS/390 and z/OS” found at

- <http://www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/TD101150>

- “WebSphere for z/OS Security Overview” found at

- <http://www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/PRS1054>

- “RACF Tips for customizing WebSphere for z/OS Version 5.0.2” found at

- <http://www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/TD101118>.

- **WebSphere Variables to control operator message routing**

This article describes how to manage operator message routing in WebSphere for z/OS V5 and can be found at:

<http://www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/TD101116>

- **DB2 UDB / JCC Connectors**

This article describes how to enable WebSphere for z/OS V5.0.2 to use the DB2 Universal JDBC Driver and can be found at:

<http://www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/TD101663>

For full details on using these connectors in the WebSphere environment, see the Redbook: “DB2 for z/OS and WebSphere: The Perfect Couple,” SG24-6319 available at www.redbooks.ibm.com.

- **Configuring and Troubleshooting the WebSphere for z/OS Version 5 HTTP Server Plugin**

This article describes how to configure and troubleshoot the WebSphere for z/OS V5 HTTP Server and can be found at:

<http://www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/PRS829>

Chapter 17. Using EIM authentication

Enterprise Identity Mapping (EIM) is an IBM @server infrastructure architecture that defines a set of services and extensions to LDAP to transform the user identity associated with a work request as it moves between systems having different user administration schemes as part of a multi-tiered application in a heterogeneous environment. EIM offers a new approach to easily manage multiple user registries and user identities in an enterprise by providing an architecture for describing the relationships between entities (such as individual users and system resources) in the enterprise and the many identities that represent them.

In z/OS V1R5, EIM has added support for the following bind types:

- Client authentication using a digital certificate over an SSL connection
- Kerberos authentication for clients and servers using a trusted third party protocol
- CRAM-MD5 password protection using a hashed password

The following sections describe our experiences deploying each bind type.

Client authentication using digital certificates

We used information from the following sources to help us set up and test EIM with client authentication using digital certificates:

- *z/OS Integrated Security Services EIM Guide and Reference*, SA22-7875
- *z/OS Integrated Security Services LDAP Server Administration and Use*, SC24-5923
- *z/OS Integrated Security Services LDAP Client Programming*, SC24-5924
- *z/OS HTTP Server Planning, Installing, and Using*, SC34-4826
- *z/OS Cryptographic Services System Secure Sockets Layer Programming*, SC24-5901

We used the System Secure Sockets Layer (SSL) gskkyman utility to generate our key databases and certificates. We know from past experience that *z/OS HTTP Server Planning, Installing, and Using* does an excellent job of explaining how to generate certificates. We used that document to assist in the generation of the certificates for EIM/LDAP, even though the HTTP server was not involved. (Basically, a certificate is a certificate, regardless of the application with which it will be used.) We then used the EIM and LDAP documentation to assist in the implementation and use of the certificates.

Resolving problems during our testing

EIM domain name missing the country attribute: The EIM domain that we had established in our previous testing did not have a country attribute in its domain name. The EIM domain name is the full distinguished name (DN) of the EIM domain. The gskkyman utility requires a domain name to contain, at a minimum, common name (cn), organization (o), and country (c). We created a new EIM domain name containing the required attributes so that our client certificates would have associated entries in the EIM domain controller. It is critical not only to insure that the domain name contains the minimum required attributes, but also that the domain name in the EIM domain controller matches the domain name in the client certificate.

Enterprise Identity Mapping

Using the documented example for creating LDAP suffix and user objects: To set up a new suffix, we used the “Example for creating LDAP suffix and user objects” in *z/OS Integrated Security Services EIM Guide and Reference*. If you use this example, it is important to make sure that a blank line exists between the object entries in the sample `ldif` file. The blank line is what signals the end of one entry and the beginning of the next. When we created the `ldif` file, the blank lines were missing and, thus, the `ldapadd` command failed.

We also did not include the entry that defines the country object (`c=us`). This is because our `slapd.conf` file only defines a suffix for both the organization and country (`o=ibm,c=us`). If we had included the entry for the country object as in the example, we would have had to add that suffix to our `slapd.conf` file. This illustrates the importance of understanding the structure of the data in your directory and how it is processed.

Testing the client authentication using digital certificates

Enabling the EIM domain controller for SSL processing: We followed the instructions in the LDAP Server documentation to enable our EIM domain controller for SSL processing. We also used the HTTP Server documentation to generate the key database and digital certificates.

Testing the client certificate: We issued the following `ldapsearch` command to verify that SSL processing was working properly:

```
ldapsearch -h ip_address -S EXTERNAL -Z -l 689 -K client_key_database
           -P client_key_database_password -V 3 -p 689 -s base -b "" "objectclass=**"
```

This worked successfully.

We then issued the following `eimadmin` command using a client certificate:

```
eimadmin -lD -d 'ibm-eimDomainName=SSL Domain,o=eimssl,c=us' -h ldaps://ip_address:689
          -K client_key_database -P client_key_database_password -S EXTERNAL
```

This also worked successfully.

Again, it is important to understand the structure of your data. Also, because the command syntax is long and complex, it is easy to make a mistake and it's not always clear from the error messages what is wrong.

Kerberos authentication

We used information from the following sources to help us set up and test Kerberos authentication:

- *z/OS Integrated Security Services EIM Guide and Reference*, SA22-7875
- *z/OS Integrated Security Services LDAP Server Administration and Use*, SC24-5923
- *z/OS Integrated Security Services LDAP Client Programming*, SC24-5924
- *z/OS Integrated Security Services Network Authentication Service Administration*, SC24-5926

We heavily relied on the LDAP documentation to set up the environment to allow EIM to bind using Kerberos. There are many different ways to set up this environment—what we describe here is but one way.

Clearing up a documentation inaccuracy

In the chapter on EIM APIs in *z/OS Integrated Security Services EIM Guide and Reference*, it states in three different places that, “To connect to an EIM domain using Kerberos information, you need to do so from a non-z/OS platform.” This statement occurs under the descriptions of the `eimListUserAccess`, `eimQueryAccess`, and `eimRemoveAccess` APIs.

The functionality to connect to an EIM domain from a z/OS platform using Kerberos information is available in z/OS V1R5. The above statement is left over from the previous release and, unfortunately, was not removed from the documentation in time for the general availability (GA) of z/OS V1R5. However, the statement will be removed from the next release of the documentation.

Testing the Kerberos authentication

Enabling the EIM domain controller for Kerberos processing: We followed the instructions in the LDAP Server documentation to enable our EIM domain controller for Kerberos processing. In addition, refer to our December 2002 edition for more information on enabling LDAP Server for use with Kerberos authentication.

Testing the EIM bind with Kerberos authentication: We used the `eimadmin` command for our testing. The EIM documentation describes the command syntax and how to bind using Kerberos authentication.

Before we could issue the `eimadmin` command, we first had to obtain a Kerberos ticket by issuing the `kinit` command, as follows:

```
kinit kerberos_principal
```

Once we had the Kerberos ticket, we issued the following `eimadmin` command specifying that we wanted to bind with Kerberos authentication:

```
eimadmin -lR -d 'ibm-eimDomainName=Domain,o=org,c=us' -r 'RACF SSL'
-h ldap://ip_address -S GSSAPI
```

Note that the `eimadmin` command does not have any bind credentials. The `-S GSSAPI` specifies to use the Kerberos ticket obtained from the `kinit` command. However, unless something is done, the Kerberos principal identified in the ticket does not have authorization to issue EIM commands. The EIM documentation does not specify how to do this. However, there are several ways to identify or associate the Kerberos principal to IDs (or, LDAP DNs) that are already authorized to issue EIM commands. We'll describe one method that we used.

Using TDBM mapping to associate a Kerberos principal to an EIM-authorized ID: We chose to use TDBM mapping to associate the Kerberos principal to an EIM-authorized DN, as described in *z/OS Integrated Security Services LDAP Server Administration and Use* under the “Identity mapping” section of the chapter on “Kerberos authentication”. To do this, we created an `ldif` file, `kerberos.ldif`, containing the following:

```
dn: cn=eim ssl administrator,o=eimssl,c=us
changetype:modify
add:x
objectclass: ibm-securityIdentities
altSecurityIdentities: KERBEROS:principal@REALM
```

We then issued the following `ldapmodify` command to update the LDAP DN:

```
ldapmodify -h ip_address -D "cn=LDAP Administrator" -w password
-f /etc/ldap/kerberos.ldif
```

Enterprise Identity Mapping

We were then able to issue **eimadmin** commands binding with Kerberos authentication. However, note that the Kerberos principal will only be able to issue those EIM commands for which its associated LDAP DN is authorized.

CRAM-MD5 password protection

We used information from *z/OS Integrated Security Services EIM Guide and Reference*, SA22-7875 to help us test CRAM-MD5 password protection:

No work is needed to set up for using CRAM-MD5 binds. All we needed to do was issue the **eimadmin** command and specify the CRAM-MD5 bind option. We did not encounter any problems using the CRAM-MD5 bind.

The following are a couple of examples of using the **eimadmin** command with the CRAM-MD5 bind option:

Example: We used the following command to list a domain:

```
eimadmin -lD -d 'ibm-eimDomainName=SSL Domain,o=eimssl,c=us' -h ldap://ip_address:389
-b 'cn=eim ssl administrator,o=eimssl,c=us' -w password -S CRAM-MD5
```

Example: We used the following command to list a registry:

```
eimadmin -lR -d 'ibm-eimDomainName=SSL Domain,o=eimssl,c=us' -r 'RACF SSL' -h ldap://ip_address:389
-b 'cn=eim ssl administrator,o=eimssl,c=us' -w password -S CRAM-MD5
```

EIM enhancements in z/OS V1R6

In z/OS V1R6 the following EIM enhancements were tested:

- “x.509 certificate registries”

We used the following documentation to help us plan and implement these enhancements:

- *z/OS Integrated Security Services EIM Guide and Reference*
- *z/OS Cryptographic Services System Secure Sockets Layer Programming*

x.509 certificate registries

First, we created an x.509 certificate registry. We used the **eimadmin** command for all of our testing.

Example: We used the following command to create the x.509 registry:

```
eimadmin -aR -h ldap://<ip address>:389 -b 'cn=EIM Administrator'
-w <password> -d 'ibm-eimDomainName=SSL Domain,o=eimssl,c=us'
-r 'Cert Maps' -y X509 -n 'Registry for Certificates'
```

Example: We used the following command to list the registry for verification:

```
eimadmin -lR -d 'ibm-eimDomainName=SSL Domain,o=eimssl,c=us'
-r 'Cert Maps' -h ldap://<ip address>:389 -b 'cn=EIM Administrator'
-w <password>
```

Result:

```
source registry: Cert Maps
registry kind: SYSTEM
registry type: X509
description: Registry for Certificates
lookups: ENABLED
policies: DISABLED
```

Testing associations

We used the following procedures to create an association using the name stored within a certificate:

Obtain the certificate in a file:

We used the System Secure Sockets Layer (SSL) gskkyman utility to retrieve an existing client certificate from an existing kdb database.

1. Issued the gskkyman command
2. Select the kdb
3. Select option 1 for Manage keys and certificates
4. Select option 'n' for the certificate to export
5. Select option 6 for Export certificate to a file
6. Select option 2 for Base64 ASN.1 DER

Example: After the certificate is in a file, we entered the EIM command to add an association from a certificate:

```
eimadmin -aA -h ldap://<ip address>:389 -b 'cn=EIM Administrator' -w <password>
-d 'ibm-eimDomainName=SSL Domain,o=eimssl,c=us' -r 'Cert Maps'
-i "eim ssl administrator" -E <certificate file name> -t admin
```

Result: This failed with the following error message because we did not create the identifier:

```
ITY4030 Service eimAddAssociation() returned error 248
ITY0025 EIM identifier not found or insufficient access to EIM data.
```

Example: Create the identifier before the association. We used the following command to create the identifier:

```
eimadmin -aI -i 'eim ssl administrator' -n 'Identifier for eim ssl admin'
-d 'ibm-eimDomainName=SSL Domain,o=eimssl,c=us' -h ldap://<ip address>:389
-b 'cn=EIM Administrator' -w <password>
```

Next, we verified by listing the identifier:

```
eimadmin -lI -d 'ibm-eimDomainName=SSL Domain,o=eimssl,c=us'
-i 'eim ssl administrator' -h ldap://<ip address>:389
-b 'cn=EIM Administrator' -w <password>
```

Result:

```
unique identifier: eim ssl administrator
other identifier: eim ssl administrator
description: Identifier for eim ssl admin
```

This worked successfully. We reissued the association command. This also worked successfully.

Note: If an identifier is used that already exists, you would not see the error on the first issuance of the command and the addition of the identifier would not be required.

Example: We listed the association for validation:

Enterprise Identity Mapping

```
eimadmin -lA -d 'ibm-eimDomainName=SSL Domain,o=eimssl,c=us'  
-i 'eim ssl administrator' -h ldap://<ip address>:389  
-b 'cn=EIM Administrator' -w <password>
```

Result: That successfully created an association from a certificate located in a file.

```
unique identifier: eim ssl administrator  
association: ADMIN  
source registry: Cert Maps  
registry type: X509  
registry user: <SDN>CN=EIMSSLADMINISTRATOR,O=EIMSSL,C=US</SDN>  
<IDN>CN=AE TEAM CA FOR JAO,OU=INTEGRATION TEST,  
O=AE TEAM,L=POK,ST=NY,C=US</IDN> <HASH_VAL>CF752E  
7699A95818799E8AC70CD6A9F8BCA0B35D</HASH_VAL>
```

Removing an association using the name stored within a certificate:

Example: We removed the association previously created, by issuing the following command:

```
eimadmin -pA -h ldap://<ip address>:389 -b 'cn=EIM Administrator'  
-w <password> -d 'ibm-eimDomainName=SSL Domain,o=eimssl,c=us'  
-r 'Cert Maps' -i 'eim ssl administrator' -E <certificate file name>-t admin
```

Example: Next, we issued the list command to verify that the association does not exist:

```
eimadmin -lA -d 'ibm-eimDomainName=SSL Domain,o=eimssl,c=us'  
-i 'eim ssl administrator' -h ldap://<ip address>:389 -b  
'cn=EIM Administrator' -w <password>
```

Result: The association was removed.

Removing an association using the -u flag: **Example:** We issued the following command to remove an association using the -u flag:

```
eimadmin -pA -h ldap://<ip address>:389 -b 'cn=EIM Administrator' -w <password>  
-d 'ibm-eimDomainName=SSL Domain,o=eimssl,c=us' -r 'Cert Maps' -i 'eim ssl  
administrator' -u '<SDN>CN=EIM SSL ADMINISTRATOR,O=EIMSSL,C=US</SDN>  
<IDN>CN=AE TEAM CA FOR JAO, OU=INTEGRATION TEST,O=AE TEAM,L=POK,ST=NY,C=US</IDN>  
<HASH_VAL>CF752E7699A95818799E8AC70CD6A9F8BCA0B35D</HASH_VAL>' -t admin
```

Result: The removal was successful.

Testing Filtering

Next, we tested the x.509 certificate filtering.

Example: We issued the following command to create a filter policy:

```
eimadmin -aY -h ldap://<ip address>:389 -b 'cn=EIM Administrator'  
-w <password> -d 'ibm-eimDomainName=Small Domain,o=EIM'  
-r 'Cert Maps' -J 'O=EIMSSL,C=US' -F 'OU=INTEGRATION TEST,  
O=AE TEAM,L=POK,ST=NY,C=US' -T 'RACF Insko' -u C00002 -t filter
```

Example: We issued the list filter command to validate the add filter command:

```
eimadmin -pY -h ldap:// ://<ip address>:389 -b 'cn=EIM Administrator' -w <password>  
-d 'ibm-eimDomainName=Small Domain,o=EIM' -r 'Cert Maps' -J 'O=EIMSSL,C=US'  
-F 'OU=INTEGRATION TEST,O=AE TEAM,L=POK,ST=NY,C=US' -t filter
```

Result: The following is returned:

```
source registry: Cert Maps  
policy type: FILTER  
filter: <SDN>O=EIMSSL,C=US</SDN><IDN>OU=INTEGRATION
```

```

TEST,O=AE TEAM,L=POK,ST=NY,C=US</IDN>
  target registry: RACF Insko
target registry user: C00002
  domain policies: DISABLED
source registry lookups: ENABLED
target registry lookups: ENABLED
target registry policies: DISABLED

```

Example: We issued the following command to delete the filter policy:

```

eimadmin -pY -h ldap://<ip address>:389 -b 'cn=EIM Administrator'
-w <password> -d 'ibm-eimDomainName=Small Domain,o=EIM' -r 'Cert Maps'
-J 'O=EIMSSL,C=US' -F 'OU=INTEGRATION TEST,O=AE TEAM,L=POK,ST=NY,C=US'
-T 'RACF Insko' -u C00002 -t filter

```

Result: The filter policy was successfully removed.

Create an x.509 certificate filter policy using a certificate

Example: We issued the following command to create a certificate filter policy:

```

eimadmin -aY -h ldap://<ip address>:389 -b 'cn=EIM Administrator'
-w <password> -d "ibm-eimDomainName=Small Domain,o=EIM" -r 'Cert Maps'
-G <certificate file name> -J O= -F OU= -T 'RACF Insko' -u C00002 -t filter

```

Example: Next, we issued the list filter command to verify using the certificate as the list criteria:

```

eimadmin -lY -h ldap://<ip address>:389 -b "cn=EIM Administrator"
-w <password> -d "ibm-eimDomainName=Small Domain,o=EIM"
-r 'Cert Maps' -G <certificate file name> -J O= -F OU= -t filter

```

Result: Here is what returned:

```

source registry: Cert Maps
                policy type: FILTER
                filter:
<SDN>=EIMSSL,C=US</SDN><IDN>OU=INTEGRATION TEST,
O=AE TEAM,L=POK,ST=NY,C=US</IDN>
  target registry: RACF Insko
target registry user: C00002
  domain policies: DISABLED
source registry lookups: ENABLED
target registry lookups: ENABLED
target registry policies: DISABLED

```

Example: To delete the filter policy using the certificate as the delete criteria, we issued the following command:

```

eimadmin -pY -h ldap://<ip address>:389 -b "cn=EIM Administrator"
-w <password> -d "ibm-eimDomainName=Small Domain,o=EIM" -r 'Cert Maps'
-T 'RACF Insko' -u C00002 -G <certificate file name> -J O= -F OU= -t filter

```

Result: The list filter was issued and nothing was returned.

Part 3. Linux virtual servers

Chapter 18. About our Linux virtual server environment	261
Chapter 19. Cloning Linux images on z/VM 5.1	263
Preparing the VM environment for the cloning system	264
Adding the FLASHCOPY command to the "Z" class	264
Defining VM userid "USER" and common DASD	264
Defining the key files on common DASD	264
USER 194 Disk	264
USER 195 Disk	265
Including the LTICPRO directory profile	265
Defining the LTICxxx directory entry	266
Setting up the LTICxxx system	266
Setting the IP and HOSTNAME on the LTICxxxx guest system	266
Verifying the setup	267
Chapter 20. Establishing security in a heterogeneous Linux server environment	269
Planning for our Linux on zSeries environment	269
Linux on zSeries network configuration	270
Linux on zSeries middleware environment	273
Existing environment	273
New middleware	273
Installing WebSphere Application Server and WebSphere Application Server Network Deployment V5.1	273
Web Servers for WebSphere Application Server and zSeries Hardware Cryptographic Acceleration	274
Configuring the IBM HTTP Server	274
Enabling HTTPS on the IBM HTTP Server	274
Configuring the Apache2 Server	274
Enabling HTTPS on the Apache2 server	274
Enabling HTTPS using hardware crypto acceleration on the Apache2 server	275
Configuring DB2 V8.1 clients on Linux on zSeries to a z/OS DB2 backend	278
Problems encountered	278
Setting SSL tunneling on in the WebSphere Application Server Edge Component Caching Proxy V5.1	279
Configuring WebSphere Application Server ND Edge Component Load Balancer V5.1	280
Problems we encountered	280
Defining Samba on Red Hat Enterprise Linux 3 Update 4	281
Installing and running Domino Mail Server V6.5.4 on Linux on zSeries	282
Installing the Domino server on Linux on zSeries	283
Installing the Domino Administrator Client on Windows	284
Starting the Domino Server on the Linux on zSeries system in Setup mode	284
Running the Setup program from the Administrator's Windows system	284
Restarting the Domino server on the Linux on zSeries system	285
Configuring the Domino Administrator system to work with the server	285
Defining clients to the server	285
Installing the Lotus Notes Client systems on additional Windows machines	285
Open source security products	286
Changing the placement of Hogwash	286

	iptables	286
	Defining the rules for the firewall between between Public LAN and	
	VLAN674.	286
	Defining the rules for the firewall between VLAN673 and VLAN672	287
	IBM security products	290
	Planning and installing Tivoli Risk Manager (TRM) Host IDs	290
	Components of our TRM installation	291
	Checking for incidents	293
	Problems we encountered	294
	Authenticating and authorizing Web transactions using Tivoli Access	
	Manager and TAM WebSeal.	295
	Problems we encountered	296
	Authenticating Linux users using RACF and LDAP on z/OS	296
	Independent service vendor (ISV) security products	297
	Installing TrendMicro's ScanMail	297
	Testing to see if it detects viruses	298
	Installing TrendMicro's ServerProtect	298
	Security testing	301
	Security: Next steps	301
	Chapter 21. Future Linux on zSeries projects	303
	Migration	303
	High availability	303
	Where to find more information	303

The following chapters describe the Linux virtual servers aspects of our computing environment.

Chapter 18. About our Linux virtual server environment

In recent years, Linux has emerged as an operating system for the enterprise. As a result, the zSeries Integration Test team has recently expanded to add a Linux virtual server arm to its overall environment, which will be used to emulate leading-edge customer environments, workloads, and activities.

This section consists of the following, including two solutions we worked on:

- “Cloning Linux images on z/VM 5.1” – We rolled our own cloning solution using what was available to us. See Chapter 19, “Cloning Linux images on z/VM 5.1,” on page 263.
- “Establishing security in a heterogeneous Linux server environment” – We’ll talk about our network configuration, middleware environment, and the security products used as well as security testing conducted. See Chapter 20, “Establishing security in a heterogeneous Linux server environment,” on page 269.
- Products we’ll talk about:
 - z/VM
 - Linux on zSeries
 - WebSphere Application Server (WAS)
 - WebSphere Application Server Network Deployment (including Edge Components)
 - Tivoli Access Manager for e-business (TAM)
 - TAM WebSEAL
 - Tivoli Risk Manager
 - TrendMicro ScanMail
 - TrendMicro ServerProtect
 - iptables
 - z/OS LDAP and z/OS RACF
 - DB2
 - Apache plus zSeries hardware cryptographic acceleration
 - and various other open source security products.
- Chapter 21, “Future Linux on zSeries projects,” on page 303 – What’s coming soon.

Chapter 19. Cloning Linux images on z/VM 5.1

Our team needed a simple system to clone Linux images running on VM that would require very little care and feeding. We have come up with a simple solution using FLASHCOPY2 and a few basic REXX EXECs and a script running on Linux. Because the test environment spans a number of VM systems, the cloning system has been designed to run across multiple VM systems using shared DASD.

This chapter outlines how we implemented a cloning solution. It is not intended to be a complete reference and installation guide but rather a description of how the system works and what is needed to implement it in your shop. The REXX EXECs and Linux scripts are not the most elegant, but they get the job done. Although there are many different ways to approach cloning of Linux systems, this method works well for us because of its simplicity and ability to span VM systems, not because it includes a robust set of features. Its focus is on provisioning a Linux distribution into existing z/VM guests. It does not dynamically create z/VM guests. The z/VM administrator must manually define the z/VM guests where the Linux distribution are provisioned and IPLed.

Assumptions: Consider these few assumptions for our test environment:

- z/VM is installed and running.
- DASD supports FLASHCOPY2
- All Master Images and LTICxxxx 201 disks reside on the same Enterprise Storage System (ESS).
- A basic understanding of the z/VM, REXX and the VM Directory is needed.
- A basic understanding of Linux and scripting is needed.

Notes:

1. All of the Linux guests are named LTICxxxx where xxxx is from 0000 to 9999. The LTIC0000 guest is reserved for building new master Linux images. You can use any naming convention you would like. However doing so requires updates in the EXEC's and scripts. In this chapter we will often refer to the Linux guest as the LTICxxxx system or LTICxxxx guest.
2. The cloning system is made up of 3 basic parts:
 - The Linux master images
 - The VM EXECs
 - The Linux setup script.

The Linux master images: The Linux master images are Linux images installed on a single 3390 Model 3 with an IP address of 192.168.70.170 and a hostname of LTIC0000. We try to have one Linux image of every supported flavor available for our users to IPL; for example, SUSE LINUX Enterprise Linux 9, SUSE LINUX Enterprise Linux 9 SP1, Red Hat EL 3, and Red Hat EL 4, just to name a few.

The VM EXECs: Each of the Linux guests across all z/VM LPARs access a shared read-only 191 disk containing the PROFILE EXEC that sets up the environment and prompts the user with a menu. From the menu the user can "IPL Linux", "Clone a new Linux image" or "Install Linux from RAM disk". It also provides the user with a few basic tools.

The Linux setup script: This script needs to be on a FTP server that the Linux master images have access to. After a Master Linux image is built, the setup scripts are installed on the Master Linux image.

Preparing the VM environment for the cloning system

The following steps were used in setting up VM in our testing:

- Adding the FLASHCOPY command to the "Z" class
- Defining VM userid "USER" and common DASD
- Defining the key files on common DASD
- Including the LTICPRO directory profile
- Defining the LTICxxx directory entry
- Setting up the LTICxxxx system
- Setting up the IP and HOSTNAME on the LTICxxxx guest system
- Verifying the setup.

Adding the FLASHCOPY command to the "Z" class

Flashcopy requires a guest to have CLASS B privileges. We defined a new class, CLASS Z, to which we added the FLASHCOPY command.

Defining VM userid "USER" and common DASD

The Cloning system requires VM userid "USER" to be defined in the z/VM directory with minidisks 194 and 195. These disks contain all common files used by the LTICxxxx automation system, and are shared by all z/VM LPARs. All LTICxxxx guests on all z/VMs will link to these shared disks.

USER 194 - This disk is linked as the 191 disk on the linux guest.
USER 195 - This is the source disk for all of the automation EXECs.

The definitions for the disks are as follows:

```
MDISK 0194 3390 1 005 VM5435 RR ALL ALL ALL
MDISK 0195 3390 1 END VM5436 RR ALL ALL ALL
```

Note: 194 and 195 disks are defined in the directory as RR to prevent one or more VM systems from accessing the volume in write mode. Doing so would corrupt data. Link the disk MR from MAINT when you need to update the volume.

Defining the key files on common DASD

Following are some of the key files that we defined for our environment. Samples of these files can be found in Appendix C, "Some of our Linux for zSeries samples, scripts and EXECs," on page 313.

USER 194 Disk

This disk is linked as the LTICxxxx 191 disk R/O.

PROFILE EXEC: This EXEC is run when the LTICxxxx guest is logged on. This EXEC performs 2 functions.

1. If the user is autologged, this EXEC will IPL the linux system.
2. If the user logs onto the LTICxxxx guest, this EXEC will call the "WELCOME EXEC"

WELCOME EXEC: This is the main menu for the LTICxxxx automated system. This EXEC will prompt the user with the following options.

1. IPL Linux on the 201 disk
2. IPL Linux on the a disk other then 201

3. Install a new Linux system
4. Copy a pre-built Linux system
5. Display my networking information
6. What can I do with a LTICxxxx system
7. EXIT

The menu calls the following EXEC's to perform the function:

- 3 - DISTRO EXEC - on USER 195 Disk
- 4 - DISTCOPY EXEC - on USER 195 Disk
- 5 - IPDATA EXEC - on USER 195 Disk
- 6 - LTIC HELPCMS - on USER 194 Disk

USER 195 Disk

This disk is linked as the LTICxxxx user's 192 disk R/O and contains the following:

DISKCOPY LIST: This file contains a list of all available Linux systems that can be cloned. This file must be updated manually by the cloning administrator. The file consists of a description of the Linux systems and the device address that the distribution is on.

DISKCOPY EXEC: This is the EXEC that will perform the copy of the Linux system. We say "COPY", not cloning because it does not perform any function other than the copy. All Linux configuration and Linux customization must be done manually after the copy has completed. The EXEC will present the user with a list of systems that can be copied. After a system is selected, the copy will begin.

DISTRO LIST: This file contains a list of all the Linux systems that can be installed using the LTICxxxx automation system. This file must be kept up to date manually by the LTIC administrator.

DISTRO EXEC: This EXEC will present the user with a list of systems that can be installed. Once the selection is made, it will load the RAM disk install/recovery system.

IPDATA LIST: This file contains a list of all the LTICxxxx IDs and the associated IP addresses.

IPDATA EXEC: This EXEC will give the user all kinds of information about it's TCPIP configuration. Device addresses that are available, IP address, and so on.

Other files found on the 195 disk: All of the files needed to start a RAM disk system are on the 195 disk. There are 3 files for each distribution. The filetypes are IMAGE, PARM and INITRD. The filenames are a unique system identifier that is mapped to the distribution in the DISTRO LIST file.

Including the LTICPRO directory profile

The following directory profile needs to be included on all VM systems using the LTICxxxx automated system. You may need to modify the profile for your environment. The goal of the LTICxxxx system is to use common devices across all VM images. This allows you to move the Linux system from one VM system or LPAR to another and not require any changes to the Linux image. This is why we put the network NIC definition in the profile, which attaches each Linux guest to the VSWITCHes "Intranet" and "Private."

```
Profile LTICPRO
CRYPTO APVIRT
IUCV ALLOW
```

```

|          SPOOL 00C 2540 READER
|          SPOOL 00D 2540 PUNCH
|          SPOOL 00E 1403 A
|          CONSOLE 009 3215 T
|          LINK MAINT 0190 0190 RR
|          LINK MAINT 019D 019D RR
|          LINK MAINT 019E 019E RR
|          LINK USER 194 191 RR
|          LINK USER 195 192 RR
|          NICDEF 9A0 TYPE QDIO LAN SYSTEM INTRANET
|          NICDEF 600 TYPE QDIO LAN SYSTEM PRIVATE

```

Defining the LTICxxx directory entry

The directory entry for each Linux guests is the same. Only the real device address of the 0201 minidisk changes. We use the DEVNO rather than the volume label for the 0201 disk because when we clone the Linux system the volume label will change. This also gives us the ability to IPL the volume on a LPAR because it is a full pack minidisk.

```

|          USER LTIC0001 999999 256M 1024M GZ
|          INCLUDE LTICPRO
|          IPL CMS PARM AUTOOCR
|          MACHINE ESA
|          MDISK 200 FB-512 V-DISK 2048000 MR
|          MDISK 0201 3390 DEVNO 5731 MR ALL ALL ALL

```

Setting up the LTICxxx system

Setting up your VM system to use the LTICxxx cloning system requires only a few steps.

1. Create the user 194 and 195 disks.
2. Copy the EXEC's from Appendix C, "Some of our Linux for zSeries samples, scripts and EXECs," on page 313 to the 194 and 195 disks.
3. Create the LTICPRO profile in your VM directory.
4. Create the LTICxxxx systems in the VM directory.
5. Define the VSWITCH networks for INTRANET and PRIVATE .
6. Build the Master Linux images on full 3390 mod 3's starting at CYL 0.
7. Update IPDATA LIST with your installations IP addresses and system names.
8. Update DISKCOPY LIST with the Master Linux images you have built.
9. Update DISTRO LIST with the RAM disk system you will support.
10. Update the Master Linux images with the files and scripts needed to customize the list system detailed in the section "Setting the IP and HOSTNAME on LTICxxxx Guest Systems".

Setting the IP and HOSTNAME on the LTICxxxx guest system

The script (lticIPsetup) sets the hostname and the IP address on the cloned Linux system after it has been IPLed. The following need to be set up on the master Linux system(s) in order for this EXEC to work.

1. The HOSTNAME of the system MUST be ltic0000
2. The IP address MUST be 192.168.70.170
3. A list of allowable IP address and Hostnames must be in the file
/etc/ip.list
4. The file /etc/run_lticIPsetup MUST exist for the script to run. This is a 0 byte file that can be created with touch. For example:
touch /etc/run_lticIPsetup

- 5. You must add the line `/etc/init.d/liticIPsetup` to one of the following for the script to run.
 - a. On SUSE Linux:
`/etc/init.d/boot.local`
 - b. On Red Hat
`/etc/rc.d/rc.local`

Verifying the setup

The following are the steps we took to verify our setup:

1. Log onto one of the predefined LTICxxxx z/VM guests.
2. Select Option 4, Copy a pre-built Linux System

Note: If you do not have FLASHCOPY installed on your DASD, DDR will do the copy.

3. After the copy is complete, select Option 1, and IPL Linux on the 201 disk. Your Linux image should IPL.

Chapter 20. Establishing security in a heterogeneous Linux server environment

The Test and Integration Center for Linux conducted an open source security study in 2003 and 2004 that used only open source security products to secure a Linux server environment. The environment being protected consisted of various IBM middleware products such as WebSphere Application Server, and DB2. The study is documented in a white paper which can be accessed here:

<ftp://ftp.software.ibm.com/eserver/zseries/misc/literature/pdf/whitepapers/gm130636.pdf>

This year, with the merging of Linux, z/VM and z/OS Integration Test activities into a common environment, we decided to expand the security study, by extending to IBM and Independent software vendor (ISV) security products. In the sections that follow, we'll provide details to our planning, environment, and the products used.

Planning for our Linux on zSeries environment

We had an existing environment of various middleware products that we wanted to secure using a set of IBM and ISV products. We started by creating a matrix of all of our existing middleware products and the security products we wanted to use, and the OS levels they supported, in order to figure out their compatibility. If a security product is to be installed on an existing Linux guest, then it must support that OS level. Tivoli Risk Manager Host Intrusion Detection system is one example. We wanted to install TRM Host IDS adapters on all our WebSphere Application Servers, and our WebSphere Application Server servers on zSeries were running on SUSE LINUX Enterprise Server 8 SP3, that meant we could only use TRM Host IDS if it also supported SUSE LINUX Enterprise Server 8 SP3. The compatibility matrix helped us see clearly what's compatible and what's not. IBM also has out a middleware on Linux matrix for each of its eServers, <http://www.ibm.com/linux/matrix/linuxmatrixhwz.html>, and this helped us as well.

When you are doing your planning, also keep in mind the various prerequisite products that many products require. They may or may not be supported on the same OS level that the product itself is. In that case, you need to think about where you can place the prerequisites so that you can use the product. Usually, the product's release notes or planning guide will provide topology information and recommended prerequisite placements.

Table 16. Middleware compatibility matrix.

Middleware product versions:	Distro installed on:	Platform:
WebSphere Application Server 5.1.0	RHEL AS 2.1 SP1, SLES8 SP3	xSeries® zSeries
WebSphere Application Server Network Deployment 5.1.0	RHEL AS 2.1 SP1, SLES8 SP3	xSeries zSeries
DB2 V8R1M0	z/OS V1R6	zSeries
DB2 Connect 8.1.0.16	RHEL AS 2.1 SP1, SLES8 SP3	xSeries zSeries
DB2 UDB 8.1.0.16	SLES8 SP3	zSeries

Table 16. Middleware compatibility matrix. (continued)

Middleware product versions:	Distro installed on:	Platform:
WebSphere Application Server Network Deployment Edge Component Caching Proxy 5.1.0	SLES8 SP2	zSeries
WebSphere Application Server Network Deployment Edge Component Load Balancer 5.1.0	SLES8 SP2	zSeries
TAM/WebSeal 5.1	SLES8 SP3	zSeries
Tivoli Directory Server 5.1	RHEL AS 2.1 SP1	xSeries
Tivole Storage Manger 5.2.0	mixed	zSeries
Security product versions:	Distro supported:	Platform:
Apache 2.0.49	SLES8 SP3	zSeries
z/OS LDAP V1R6	z/OS V1R6	zSeries
TrendMicro ScanMail 2.6	SLES8 SP3	zSeries
TrendMicro ServerProtect 1.3	SLES8 SP3	zSeries
TRM 4.2	SLES8 SP3	zSeries

Linux on zSeries network configuration

The intent of Figure 50 on page 271 is to show the various network connections and flow of traffic.

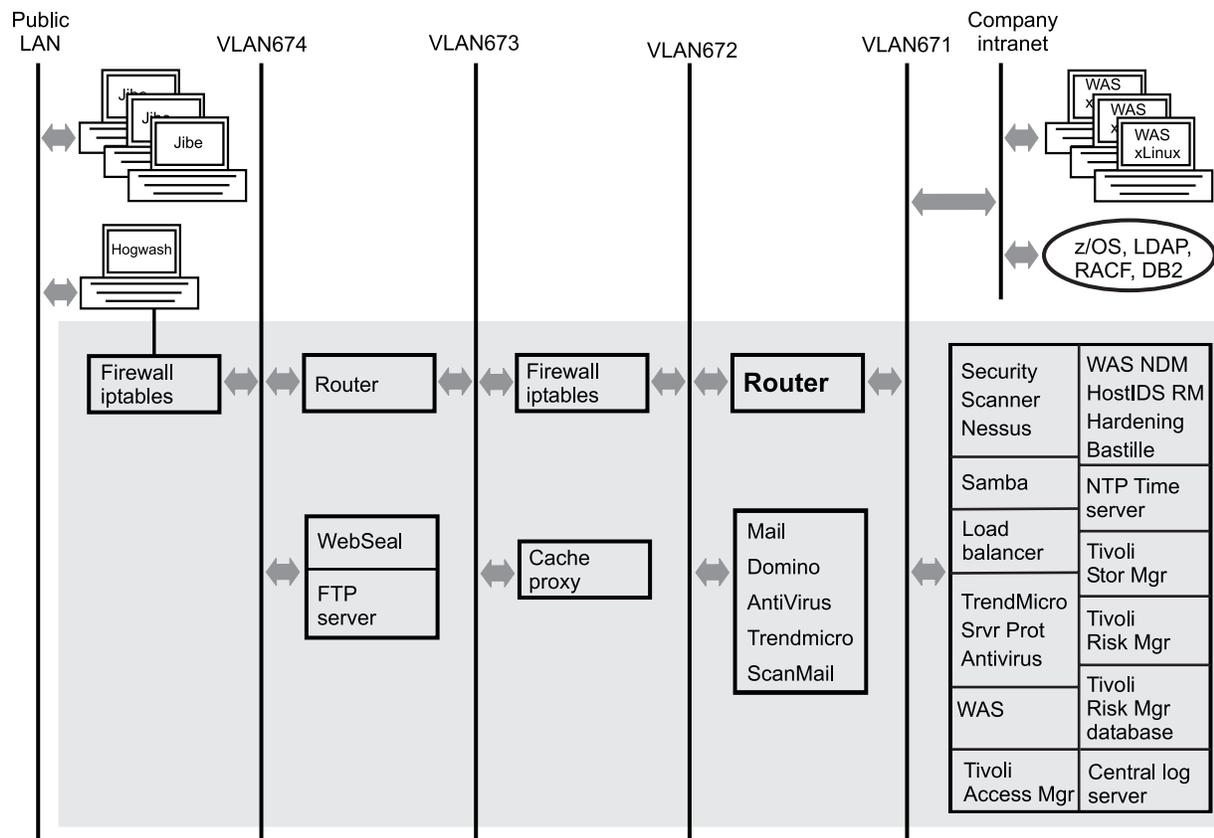


Figure 50. Linux on zSeries network configuration.

The shaded area contains our Linux guests, which are running on a single VM system on a z900 processor. The arrows are used to distinguish which network segment the guests are running on. xSeries Linux systems are represented by the PC icons. There is also a connection to z/OS backend applications through the company intranet. The primary flow of traffic goes through a firewall, a router, another firewall and another router, all of which have IP_forwarding enabled to allow the flow of packets. Which distribution you are running will determine where this is enabled. The router images are configured to compensate for the lack of certain originally-planned elements of the configuration that were omitted because of time constraints. For example, we ran out of time before configuring a mail server proxy between VLAN674 and VLAN673, so the router was required to enable mail traffic to flow from the public LAN to the Domino server.

We encountered the following issues while setting up our network environment over multiple LANs.

- **VSWITCH definitions and grants:** Initially, we were able to ping anything coupled to the VSWITCH from anything else coupled to that VSWITCH but when the traffic went out on the wire, the physical switch wouldn't allow VLAN672 tagged frames through, so the clients never saw them. Nothing outside could ping a Linux guest on that VSWITCH. The following were our definition and grant statements:

```
DEFINE VSWITCH PRVV72 RDEV 1108 VLAN 672 PORTTYPE ACCESS PORT PRVVLAN
SET VSWITCH PRVV72 GRANT LITDOM01
```

The VLAN keyword on DEFINE VSWITCH defines the default VLAN ID active in the switch. CP will associate this VLAN ID with the untagged frames that it sends and receives. Because our z/VM guests were not VLAN-aware, and because we

didn't specify a VLAN ID on our SET VSWITCH command for each guest, they were sending untagged frames which the VSWITCH then associated with a default VLAN ID of 672. At the same time, our xSeries systems were also not VLAN-aware, so they were sending untagged frames, but the physical switch in our network was associating those frames with a default VLAN ID of 1. The two defaults did not match. The switch default is usually VLAN 1 but you have to ask the physical switch maintainers to be certain. The PORT option was not required and was removed. We corrected our definition and grant statements as follows:

```
DEFINE VSWITCH PRVV72 RDEV 1108 VLAN 1 PORTTYPE ACCESS
SET VSWITCH PRVV72 GRANT LITDOM01 VLAN 672
```

We went forward with the following definitions (not all grants shown):

```
DEFINE VSWITCH PRVV71 RDEV 1104 VLAN 1 PORTTYPE ACCESS
SET VSWITCH PRVV71 GRANT LITDAT01 VLAN 671
```

```
DEFINE VSWITCH PRVV72 RDEV 1108 VLAN 1 PORTTYPE ACCESS
SET VSWITCH PRVV72 GRANT LITDOM01 VLAN 672
```

```
DEFINE VSWITCH PRVV73 RDEV 110C VLAN 1 PORTTYPE ACCESS
SET VSWITCH PRVV73 GRANT LITCP01 VLAN 673
```

```
DEFINE VSWITCH PRVV74 RDEV 1110 VLAN 1 PORTTYPE ACCESS
SET VSWITCH PRVV74 GRANT LITDCS3 VLAN 674
```

- **Inability to communicate from the Linux on zSeries systems on VLAN671 to the xSeries for Linux systems running on the company intranet:** In the network diagram, you will notice the **Router** that is **bold** between VLAN671 and VLAN672. This image was defined as primary router on VSWITCH PRVV71 and VLAN-aware. VSWITCH PRVV71 was also defined as primary router on its OSA. With these definitions, we could not communicate outside our configuration.

To resolve this issue, a separate VSWITCH was created and the Router image was granted to it. This VSWITCH had to be defined as the primary router and VLAN unaware. This VSWITCH also used a different OSA adapter that was connected to a switch port which did not send VLAN tagged frames. The following shows how the VSWITCH was created:

```
DEFINE VSWITCH IT71 RDEV 700 PRIROUTER PORT IT71
SET VSWITCH IT71 GRANT LITROUT1
```

```
#cp q vswitch it71
VSWITCH SYSTEM IT71      Type: VSWITCH Connected: 1      Maxconn: INFINITE
  PERSISTENT RESTRICTED  PRIROUTER              Accounting: OFF
  VLAN Unaware
  State: Ready
  IPTimeout: 5           QueueStorage: 8
  Portname: IT71        RDEV: 0700 Controller: TCPIP   VDEV: 0700
```

- **Inability to communicate to our network from the Public LAN:** The packets first go through Hogwash, which is an inline packet scrubber that runs on an xSeries box and is transparent to the network. Next is the Firewall between the Public LAN and VLAN674. We have dedicated a real OSA device in the VM directory to this Linux Firewall guest (which is on VLAN674). This OSA device was cabled directly into the xSeries Hogwash box, not into a switch. This needed to be defined with the primary router option. The default gateway on the xSeries systems needed to point to the Firewall. These two changes enabled the Public LAN to communicate with the rest of our network:

1. To update definitions in /etc/chandev.conf we did the following:

```
noauto;qeth1,0x1400,0x1401,0x1402;add_parms,0x10,0x1400,0x1402,\
portname:CHP07,primary_router
```

2. To set the default gateway on our xSeries systems we ran:

```
route add def gw 192.168.75.252
```

Linux on zSeries middleware environment

The following topics describe the Linux on zSeries middleware test environment.

Existing environment

We had two WebSphere Application Server clusters serving a test application called Trade3. One cluster resided on Linux on zSeries, and another on xSeries. Redundant clusters enhance overall availability, and we chose to spread the clusters across two different hardware platforms to add more variety to the test environment.

We used the WebSphere Application Server Network Deployment Edge Component Load Balancer to balance the load between the two clusters, and the cluster WebSphere Application Server Network Deployment Manager to balance the load among its cluster members. WebSphere Application Server Network Deployment Edge Component Caching Proxy was used to tunnel SSL connections from the Tivoli Access Manager WebSEAL component to the Load Balancer.

New middleware

We also set up Domino mail server to provide mail services, and a Samba server for file serving purposes. The FTP server was set up for anonymous file transfer and can be used for transferring certain files from our Samba server.

We'll talk about the WebSphere Application Server components, Samba, and Domino in this section. For WebSEAL/TAM configuration See "IBM security products" on page 290.

Installing WebSphere Application Server and WebSphere Application Server Network Deployment V5.1

Installing WebSphere Application Server and WebSphere Application Server Network Deployment V5.1 requires that you install V5.0 first and then use the update installer to install the fixpack. We followed the installation guide in the WebSphere Application Server Info Center at

<http://publib.boulder.ibm.com/infocenter/ws51help/index.jsp>

and the readme file that comes with the fixpack to install WebSphere Application Server V5.1.

When you have installed WebSphere Application Server and WebSphere Application Server Network Deployment V5.1, you must federate the WebSphere Application Server nodes to the WebSphere Application Server Network Deployment Manager to form a cluster. We used the following command for the federation to the WebSphere Application Server Network Deployment V5.1 node, using our nodehostname of litwas04.ltic.pok.ibm.com:

```
# $WAS_HOME/bin/addNode.sh litwas04.ltic.pok.ibm.com
```

We installed Trade3 on WebSphere Application Server Network Deployment Manager and synchronized with cluster members. We configured one cluster on zSeries and another on xSeries.

Web Servers for WebSphere Application Server and zSeries Hardware Cryptographic Acceleration

We used one web server for each cluster. For the xSeries cluster, we used the IBM HTTP Server that came with WebSphere Application Server. For the zSeries cluster, we built Apache2 version 2.0.49.

Configuring the IBM HTTP Server

We used the IBM HTTP Server that came with WebSphere Application Server, and put it on a separate system. We copied the plugin-cfg.xml file from the WebSphere Application Server cluster (any member from the cluster will have the same plugin-cfg.xml file) and pointed to it in the httpd.conf file of the web server, as follows:

```
LoadModule ibm_app_server_http_module \
/opt/WebSphere/AppServer/bin/mod_ibm_app_server_http.so \
WebSpherePluginConfig /opt/WebSphere/AppServer/config/cells/plugin-cfg.xml
```

Enabling HTTPS on the IBM HTTP Server

We used the following steps to enable HTTPS on the IBM HTTP Server:

- Create a key database named plugin-key.kdb and store it in /opt/WebSphere/AppServer/etc
- Generate a self-signed certificate with the label 'WebSphere Plugin Key'
- Add the following changes to the IHS conf file /opt/IBMHttpServer/conf/httpd.conf

```
LoadModule ibm_ssl_module libexec/mod_ibm_ssl_128.so
<VirtualHost *:443>
DocumentRoot /opt/IBMHttpServer/htdocs/en_US
    SSLEnable
    SSLClientAuth none
    SSLServerCert WebSphere Plugin Key
    Keyfile /opt/WebSphere/AppServer/etc/plugin-key.kdb
</VirtualHost>
```

- Restart the web server normally.

```
/opt/IBMHttpServer/bin/apachectl restart
```

Configuring the Apache2 Server

For the zSeries cluster, we built Apache2, version 2.0.49, on a SUSE LINUX Enterprise Server 8 SP3 system, and took advantage of the PCICA crypto card on our z990 hardware. At the time of this test, we used the crypto card on the z990 since there were no crypto cards available for us on the z900. The parameters to insert in the httpd.conf file is a little different for Apache2 than IBM HTTP Server:

```
LoadModule was_ap20_module \
/usr/local/apache2.0.49_susessl7c/mod_was_ap20_http.so \
WebSpherePluginConfig /usr/local/apache2.0.49_susessl7c/plugin-cfg.xml
```

Enabling HTTPS on the Apache2 server

We used the following steps to enable HTTPS on the Apache2 server:

- Create a self-signed certificate using openssl.
- Edit /etc/ssl/openssl.cnf and change so stateOrProvinceName is optional:
stateOrProvinceName = optional
- Issue a certificate request for a certificate which will be one of our Certificate Authorities (CA):

```
litzweb:/etc/ssl # openssl req -config openssl.cnf -new -nodes -keyout
lit.key -out lit.csr
Using configuration from openssl.cnf
Generating a 1024 bit RSA private key
```

```

.....+++++
..+++++
writing new private key to 'lit.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----

```

```

Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:IBM
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:litzweb.ltic.pok.ibm.com
Email Address []:

```

```

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:password
An optional company name []:

```

- **Encrypt the key:**

```

litzweb:/etc/ssl # openssl rsa -in lit.key -des3 -out litencrypt.key
read RSA key
writing RSA key
Enter PEM pass phrase: password
Verifying password - Enter PEM pass phrase: password

```

- **Sign the certificate:**

```

litzweb:/etc/ssl # openssl x509 -in lit.csr -out lit.crt -req -signkey
litencrypt.key -days 999
Signature ok
subject=/C=US/ST=Some-State/O=IBM/CN=litzweb.ltic.pok.ibm.com
Getting Private key
Enter PEM pass phrase: password

```

- **You can now use the encrypted key database and certificate in Apache. Edit the following lines in httpd.conf to reflect the new key database and certificate:**

```

SSLCertificateFile /etc/ssl/lit.crt
SSLCertificateKeyFile /etc/ssl/litencrypt.key

```

- **Start Apache2 with SSL:**

```

litzweb:/usr/local/apache2.0.49_susessl7c/bin # ./apachectl startssl
Apache/2.0.49 mod_ssl/2.0.49 (Pass Phrase Dialog)
Some of your private key files are encrypted for security reasons.
In order to read them you have to provide us with the pass phrases.

```

```

Server litzweb.ltic.pok.ibm.com:443 (RSA)
Enter pass phrase: password

```

```

Ok: Pass Phrase Dialog successful.

```

- **Access to the Apache web server at <https://192.168.71.119> can be established.**

Enabling HTTPS using hardware crypto acceleration on the Apache2 server

The following are the details on the crypto hardware used for the test:

- PCICA: (PCI Cryptographic Accelerator):
 - Provides support for clear keys
 - Feature code 0862
 - Only available if you have CP Assist for Cryptographic Functions feature

- | – Available on zSeries 990 (can support a max of 12 PCICAs, each feature
- | code has 2 coprocessors)
- | – Enables max SSL performance

| We used the following steps to enable HTTPS using hardware crypto acceleration
| on the Apache2 server:

- | • Ensure the correct levels of openssl and libica are installed:

```
| openssl-z990-0.9.7c
| libica-z990-1.3.4-3
```

- | • Ensure the libraries in /usr/lib are linked to the correct libraries:

```
| sles8bas:/usr/lib # ln -s /opt/openssl-z990/lib/libcrypto.a libcrypto.a
| sles8bas:/usr/lib # ln -s /opt/openssl-z990/lib/libcrypto.so libcrypto.so
| sles8bas:/usr/lib # ln -s /opt/openssl-z990/lib/libssl.a libssl.a
| sles8bas:/usr/lib # ln -s /opt/openssl-z990/lib/libssl.so libssl.so
| sles8bas:/usr/lib/ # mv libica.so libica.so.OLD
| sles8bas:/usr/lib/ # ln -s /opt/libica-z990/lib/libica.so libica.so
```

- | • Change the link for openssl in /usr/include:

```
| sles8bas:/usr/include # ln -sf /opt/openssl-z990/include/openssl openssl
```

- | • Run the openssl command to verify if ibmca is a supported engine. This is the
| support required to exploit crypto acceleration for RSA algorithms used for SSL
| on PCICA card.

```
| sles8bas:/opt/openssl-z990/bin # ./openssl engine
| (dynamic) Dynamic engine loading support
| (cswift) CryptoSwift hardware engine support
| (chil) nCipher hardware engine support
| (atalla) Atalla hardware engine support
| (nuron) Nuron hardware engine support
| (ubsec) UBSEC hardware engine support
| (aep) Aep hardware engine support
| (ibmca) Ibmca hardware engine support
| (sureware) SureWare hardware engine support
| (4758cca) IBM 4758 CCA hardware engine support
```

- | • Compile/build Apache 2 to use it with crypto.

- | – Downloaded the source from <http://www.apache.org>

```
|       httpd-2.0.49.tar.gz
```

- | – Run the following commands to build Apache 2:

```
| sles8bas:~/httpd-2.0.49 # export CPPFLAGS="-DSSL_EXPERIMENTAL_ENGINE"
| sles8bas:~/httpd-2.0.49 # ./configure --prefix=/usr/local/apache2.0.49_susess17c --enable-mods-shared=all --with-
| ssl=/opt/openssl-z990 --enable-ssl --enable-rule=SSL_EXPERIMENTAL
| sles8bas:~/httpd-2.0.49 # make
| sles8bas:~/httpd-2.0.49 # make install
```

- | – Use the same steps from the previous section located on page 275 in
| generating an encrypted key database and certificate.

```
|       Key database name = private.key
|       Certificate name = private.crt
```

- | – Edit /usr/local/apache2.0.49_susess17c/conf/ssl.conf:

- | 1. Include private.key & private.crt in conf:

```
|       SSLCertificateFile /usr/local/apache2.0.49_susess17c/conf/private.crt
|       SSLCertificateKeyFile usr/local/apache2.0.49_susess17c/conf/private.key
```

- | 2. Add the following under '*SSL Global Context*':

```
|       SSLCryptoDevice ibmca
```

- | – Ensure that the correct hostname is defined in

```
|       /usr/local/apache2.0.49_susess17c/conf/ssl.conf:
```

```
|       # General setup for the virtual host
|       DocumentRoot "/usr/local/apache2.0.49_susess17c/htdocs"
|       ServerName sles8bas.pdl.pok.ibm.com:443
```

```
#ServerAdmin you@example.com
ErrorLog /usr/local/apache2.0.49_susess17c/logs/error_log
TransferLog /usr/local/apache2.0.49_susess17c/logs/access_log
```

- Load the z90crypt module. Add the following text into a bash script (z90crypt_load):

```
#!/bin/sh -
#####/
## */
## s390 only */
## */
## Copyright (c) IBM Corporation 2001 */
## Licensed Material - Program Property of IBM */
## All rights reserved */
## Licensed under the IBM Public License (IPL) */
## */
## Script to be used to load device driver z90crypt. */
## */
#####/
module="z90crypt"
device="z90crypt"
group="root"
mode="666"

# invoke insmod with all arguments we got
/sbin/insmod -f -m $module $* >z90crypt.map || exit 1

major=`cat /proc/devices | awk "\\$2==\"$module\" {print \\$1}"`

# Remove stale nodes and replace them, then give gid and perms
# create unrouted device (minor b'00000000' decimal 0)
rm -f /dev/${device}
mknod /dev/${device} c $major 0
chgrp $group /dev/${device}
chmod $mode /dev/${device}
```

- Run the script:

```
./z90crypt_load
```

- Check the hardware response of the crypto driver module. This will show you the status of the card and whether you have any open handles, requests, or pending counts:

```
sles8bas:~ # cat /proc/driver/z90crypt

z90crypt version: 1.1.2
Cryptographic domain: 4
Total device count: 1
PCICA count: 1
PCICC count: 0
requestq count: 0
pendingq count: 0
Total open handles: 0

Mask of online devices: 1 means PCICA, 2 means PCICC
0000000100000000 0000000000000000 0000000000000000 0000000000000000

Mask of waiting work element counts
0000000000000000 0000000000000000 0000000000000000 0000000000000000
```

- Start Apache2 with SSL, crypto enabled:

```
sles8bas:/usr/local/apache2.0.49_susess17c/bin # ./apachectl startssl
```

- When Apache 2 is started, a “handle” is open to z90crypt. The z90crypt status display shows one open handle:

```
sles8bas:/usr/local/apache2.0.49_susess17c/bin # cat /proc/driver/z90crypt
cat /proc/driver/z90crypt
```

```
z90crypt version: 1.1.2
Cryptographic domain: 4
Total device count: 1
PCICA count: 1
PCICC count: 0
requestq count: 0
pendingq count: 0
Total open handles: 1 <===== Once you bring up Apache with SSL, this will be 1
```

```
Mask of online devices: 1 means PCICA, 2 means PCICC
0000000100000000 0000000000000000 0000000000000000 0000000000000000
```

```
Mask of waiting work element counts
0000000000000000 0000000000000000 0000000000000000 0000000000000000
```

- Test the crypto card by running heavy SSL handshake workloads against it. Check for requests that are queued:

```
requestq count: 0
```

Configuring DB2 V8.1 clients on Linux on zSeries to a z/OS DB2 backend

The DB2 database used by our WebSphere Application Server clusters resided on z/OS. The database and its objects were created for us by the z/OS DB2 administrator. All tables and indexes are owned by DB2LIT. The following are key information we used in configuring DB2 clients on zSeries for Linux:

- Database Name – DBLNXT3
- IP address – 192.168.25.36
- Port – 446
- Database Location name - USIBMT6PETDB2

We have DB2 Connect EE V8.1 FP6 installed on WebSphere Application Server Network Deployment Manager and every WebSphere Application Server node within the cluster. On every DB2 Connect instance (db2inst1) or wherever DB2 Connect EE is installed, we needed to define the following catalog entries in order to communicate with DB2 on z/OS.

```
db2 catalog tcpip node DBLNXT3 remote 192.168.25.36 server 446
db2 catalog dcs db DBLNXT3 as USIBMT6PETDB2
db2 catalog database DBLNXT3 as DBLNXT3 at node DBLNXT3 authentication \ DCS
```

To test if the connection works, we tried a simple DB2 connect command like the following:

```
db2 connect to DBLNXT3 user db2lit using password
```

Problems encountered

Following are some of the problems we encountered:

1. When trying to connect to the database DBLNXT3, we received the following error:

```
db2lit@litwas01:~> db2 connect to DBLNXT3 user db2lit
Enter current password for db2lit:
SQL30082N Attempt to establish connection failed with security reason "15"
("PROCESSING FAILURE"). SQLSTATE=08001
```

The userid db2lit did not exist on z/OS. Once we created it, we were able to connect.

2. The following are two separate problems with the same solution:

- Connecting to the database (DBLNXT3) using the db2 connect command was possible, however, when we tried executing a test connection in the data source from the WebSphere Application Server console, we encountered the following error:

```
[4/19/05 18:41:35:513 EDT] 34923e0b DataSourceCon E DSRA8040I: Failed
to connect to the DataSource. Encountered : java.lang.Exception:
COM.ibm.db2.jdbc.DB2Exception: [IBM][CLI Driver] SQL5042N One of the
communication protocol server support processes failed to start up.
```

- When populating the trade3 database through the trade3 application, we received the following exception in
<WAS_HOME>/log/TradeCluster[x]/SystemOut.log:

```
Message:TradeConfigServlet.service(...) \
Exception trying to perform action=buildDB
Exception details: com.ibm.websphere.command.CommandException: com.ibm.db2.jcc.a.i
is not serializable.
```

To fix both problems, we defined a synchronous point manager (SPM_NAME) everywhere that DB2 Connect EE is installed. We needed to define the shortname of the host where the DB2 Connect is being run –

litwas01.ltic.pok.ibm.com

Example: Let's say DB2 Connect is running on host
litwas01.ltic.pok.ibm.com. From your DB2 instance (db2inst1), issue the following command to update the synchronous point manager name (SPM_NAME) in dbm cfg:

```
db2 update dbm cfg using SPM_NAME litwas01
```

We had to do this for all DB2 Connect EE instances.

Setting SSL tunneling on in the WebSphere Application Server Edge Component Caching Proxy V5.1

WebSphere Application Server Network Deployment Edge Component Caching Proxy provides functionality that caches static web content as well as dynamic content generated by the WebSphere Application Server. Tivoli Access Manager (TAM) WebSEAL normally acts as a reverse Web proxy by receiving HTTP/HTTPS requests from a Web browser and delivering content from its own Web server or from junctioned back-end Web application servers. Requests passing through TAM WebSEAL are evaluated by the TAM authorization service to determine whether the user is authorized to access the requested resource. In our scenario, a SSL proxy junction was created on the TAM WebSEAL server to deliver post-authorized requests to the back-end WebSphere Application Server cluster. In this case, the Caching Proxy acts as a SSL proxy that tunneled SSL web traffic from WebSEAL to the WebSphere Application Server cluster. In order to set SSL Tunneling on in the Caching Proxy, edit `ibmproxy.conf` with the following parameters:

```
Enable CONNECT
SSLTunneling ON
```

Start the Caching Proxy normally:

```
/etc/init.d/ibmproxy start
```

See “Authenticating and authorizing Web transactions using Tivoli Access Manager and TAM WebSeal” on page 295 for more details on setting up WebSEAL.

Configuring WebSphere Application Server ND Edge Component Load Balancer V5.1

With WebSphere Application Server ND Edge Component Load Balancer V5.1, it is recommended that you use the latest version of Java with Load Balancer. We are using the following Java version:

```
litlb01:~ # java -version
java version "1.4.2"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.4.2)
Classic VM (build 1.4.2, J2RE 1.4.2 IBM build cx390142sr1a-20050209
(JIT enabled: jitc))
```

Our Load Balancer configuration includes two WebSphere Application Server clusters, each with a different Web server:

- Apache2 2.0.49 on zSeries (IP: 192.168.71.119)
- IBM HTTP Server 1.3.28 on xSeries (IP: 192.168.71.32)

We created a Load Balancer cluster that encompassed both WebSphere Application Server clusters with the IP address 192.168.71.98. We created a `lb.config` script that ran all the Load Balancer commands to setup the cluster. You must make sure that `dsserver` is running before running the script. `dsserver` handles requests from the command line to various Load Balancer components. To start `dsserver`, type:

```
dsserver start
```

Here's our `lb.config` script:

```
dscontrol executor start
dscontrol set loglevel 1
dscontrol cluster add 192.168.71.98
dscontrol cluster set 192.168.71.98 proportions 49 50 1 0
dscontrol executor configure 192.168.71.98 eth0 255.255.255.0
dscontrol port add 192.168.71.98:443 reset no
dscontrol server add 192.168.71.98:443:192.168.71.32 address 192.168.71.32
dscontrol server add 192.168.71.98:443:192.168.71.119 address 192.168.71.119
dscontrol manager start manager.log 10004
dscontrol advisor start Http 443 Http_443.log
```

As you can see, we only allowed HTTPS (port 443) connections to the cluster address.

Problems we encountered

Here are some of the problems we encountered:

- Initially, we were using JDK 1.3.1, but Load Balancer was not functioning correctly. When using JDK 1.3.1, the manager and advisor reports from the cluster were not being returned. Once we updated to JDK 1.4.2, the reporting mechanism worked as expected.
- When accessing our LB cluster, it would only load balance to the xSeries Web server and never to zSeries. We found that this is a known and documented problem in the *Load Balancer's User Guide*, also known as the 'ARP problem'. To resolve the problem, we needed to issue this command once on LB as well as on every server being load balanced (in our case, the two Web servers):

```
sysctl -w net.ipv4.conf.lo.hidden=1 net.ipv4.conf.all.hidden=1
```

Every time the systems are rebooted, we had to issue this command again. We recommend you put this in a startup script so that you don't have to manually type the command on every reboot.

Defining Samba on Red Hat Enterprise Linux 3 Update 4

Samba is a file serving system that is useful if you have Windows systems and Linux on zSeries systems in the same operating environment. It allows Windows users to attach Linux on zSeries file systems as network drives. It also allows Linux users to access those files. Samba is part of the Linux on zSeries distributions and is available for use if it is configured in the system. If it is not already configured, it can be configured using rpm facilities.

We found that it was installed on our Red Hat Enterprise Linux-3 (RHEL3-update 4) system. We decided to use it as a file server for various Linux on zSeries distributions of operating systems and packages. To verify that Samba was installed we ran the following command:

```
rpm -qa |grep sam
```

Here are the results:

```
[root@litsmb01 root]# rpm -qa |grep sam
samba-common-3.0.7-1.3E.1
samba-common-3.0.7-1.3E.1
samba-3.0.7-1.3E.1
redhat-config-samba-1.0.16-2
samba-client-3.0.7-1.3E.1
samba-3.0.7-1.3E.1
samba-swat-3.0.7-1.3E.1
```

We then defined the file systems to Samba that would be used. This required updating the Samba configuration files in the /etc directory.

1. Copy the /etc/samba/smb.conf to /etc/samba/smb.conf.orig to save it

```
cp /etc/samba/smb.conf /etc/samba/smb.conf.orig
```

2. Add the following lines to the end of the file /etc/samba/smb.conf

```
# Created for I/T testing
```

```
[pubmw]
comment = Download images for install/updates of middleWare
path = /pub/mw
```

```
[pubdistro]
comment = Download images install/updates of Base systems
path = /pub/distro
```

This creates two shared filesystems under Samba, pubmw and pubdistro, which are located at the directories specified by the respective path statements. The share name is limited in length. We first named pubmw, pubmiddleWare, but the system did not recognize it so we renamed it pubmw, and that was accepted.

We then defined the users that would have access to the shared file systems.

To do this we used the command smbpasswd , which defines the users to the Samba system and establishes a Samba password for access. The process is repeated for each user:

```
[root@litsmb01 etc]# smbpasswd -a usera
New SMB password:
Retype new SMB password:
startsmbfilepwent_internal: file /etc/samba/smbpasswd did not exist.
File successfully created.
Added user usera.
[root@litsmb01 etc]# smbpasswd -a samtest1
New SMB password:
Retype new SMB password:
Added user samtest1.
[root@litsmb01 etc]#
```


Under the “Installation” chapter we found that, although the server can be on Linux on zSeries, you need Windows machines for the Domino Administrator and the Lotus Notes clients. We secured three Windows machines for our installation running Windows 2000 Professional.

The installation process is a multi-step operation that requires:

1. Installation of the server code on the Linux on zSeries system
2. The definition of an Administrator to run the server and a group for notes (We defined a user “Domino Admin” with Linux userid of “domad1” on the Linux on zSeries system for this purpose.)
3. Installation of the Domino Administrator Client package on a Windows system
4. Starting the Server Setup program on the Linux on zSeries server
5. Starting a Domino Setup program on the Windows machine where you configure the server by the remote process
6. Restarting the Domino Server on the Linux on zSeries system
7. Configuring the Lotus Domino Administrator system on the Windows machine to work with the server
8. Defining clients to the server and the client systems
9. Installing the Lotus Notes Client systems on the additional Windows machines.

When the system was up and running we exchanged a few messages between the users to verify that messages were being transmitted.

Very early in the process, we realized that each time we had to do something on the server directly we needed to log on the Linux on zSeries machine as the administrator and move to the directory containing the Lotus Notes data. In our case this directory was /notesdata, and we issued a command residing in the lotus program directory defined at /opt/lotus/bin. So we added the following line to the .bashrc file for the administrator:

```
export PATH=$PATH:/opt/lotus/bin
cd /notesdata
```

and this made it easier to issue commands.

Installing the Domino server on Linux on zSeries

The installation of the Domino server was relatively easy but we did a few things in preparation. The Administrator’s Help website is very useful to make the estimates of system requirements and names and addresses needed in the installation.

We used the SUSE YAST tool to do the following:

1. Define a Domino group to Linux on zSeries (“notes”)
2. Define a Domino administrator (“Domino Admin” with user id “domad1”)

We then defined the following directories:

1. Domino Data Directory to contain the notes data (“/notesdata”). This will hold all the data related to the server
2. Domino program directory (“/opt”) (the default directory and it was already defined).
3. A directory to hold the installation package and package contents. We defined a directory /root/domino and placed the package in that directory.

The first stage of the Domino installation and setup was done by “root” on the Linux on zSeries system. Later operation of the server was done by the Administrator

(domad1). You needed to tell the installation program about the Administrator, and the name of the Domino Data directory. We downloaded the Domino installation package (c82buna.tar) to /root/domino and untarred it (tar -xvf c82buna.tar) producing a directory called zseries. We changed to that directory and started the installation session by issuing the command (./install). A series of panels was presented where we identified what to install (components), where the Domino Data directory was located, the administrator's userid, etc. When complete, the terminating message directed us to the next step.

We found the installation step can be rerun multiple times without having to uninstall, so, if errors were made at this time, it was easy to correct them.

Installing the Domino Administrator Client on Windows

A Domino server on a Linux on zSeries box is controlled by an administrator running on a Windows machine. There was no local Administrator client package for Linux on zSeries at the time of this test. Thus we needed to install the Domino Administrator Client package. This also was relatively easy to accomplish, as follows:

1. Downloaded the package to the Windows machine
2. Ran the installation Wizard to install the package.

In preparation for using the Administrator Client, we made sure we had a method to transfer files between the Windows machine and the Linux on zSeries machine. We used the winscp3 program to download the client ID files from the server. The winscp3 freeware is available from <http://winscp.net>. It is an open source SFTP/SCP client for Windows.

The installation program itself is an installation wizard that provides a GUI interface to make it easy to install. The key thing to note is that when it asks for what to install, we needed to pick everything. We needed the "Domino Designer" and "Domino Administrator" which are not part of the default packages installed.

Starting the Domino Server on the Linux on zSeries system in Setup mode

After installing the Domino Administrator Client on the Windows machine we needed to run the Domino server setup program to complete the installation of the server. The controlling side of the setup program is run from the Windows machine but the server itself must be started in setup mode on the Linux on zSeries system.

Here is where we first needed the administrator. We logged on to the Linux on zSeries system as the Domino Administrator (domad1), moved to the Domino Data directory (/notesdata) and issued the command to start the server in setup mode. Since we set up the .bashrc file earlier (to add the /opt/lotus/bin path to the PATH variable) all we needed to do was issue server -listen. The server started up in setup mode (listening for a call from the Windows machine).

Running the Setup program from the Administrator's Windows system

We returned to the Windows machine and started the "Remote Server Setup" (Start ->Programs->Lotus Applications->Remote Server Setup). A GUI interface was started and stepped us through the setup process.

We set up the server as a stand alone mail server. The only tricky part about this process was that we wanted to run the Server Load Utility to produce a workload on the server for testing purposes. The Server Load Utility requires that the organization name be the same as the domain name. We did not realize this until

later and had to reinstall the Domino Server and Client to run the test environment. It is in the setup program that the name of the server, the name of the domain, the administrator id, and the beginning of the security levels were established.

Restarting the Domino server on the Linux on zSeries system

From this point on all operations were done under the Domino Administrator's userid on the Linux on zSeries system. When the setup program completed, the server was shut down. We needed to get it started in normal mode. This was done by logging on as the administrator and going to the Domino Data directory and issuing the command `server`.

The server started and the command did not return to the command line but continued and acted as the administrator's console. Progress messages and error messages were displayed. Later when the server was shut down, it was from this session that the `quit` subcommand is issued to terminate the server.

Configuring the Domino Administrator system to work with the server

After the setup completed we went back to the Windows machine and started the Lotus Domino Administrator to configure the Windows machine as the administrator's system. It is here that we defined the server we would work with. It was important to realize that this step defined what server the administrator would work with and not the server itself. It is in this step that we downloaded the `admin.id` file from the server (it was located in the `/notesdata` directory) to the Windows system.

Defining clients to the server

After configuring the administrator system, we defined some users. It was important to realize that there were two independent ways to make a user known to the server. The first method registered the user to the system, defined the mail database (`username.nsf`) on the server, and defined the user id file on the server (`username.id`). This method also added the user to the server directory (`names.nsf`) on the server. A second method only added a user name to the server directory. In both methods we logged on to the Lotus Domino Administrator program and clicked on the People & Groups tab. The first method used the People tab on the right and the Register pulldown menu. The second method used the People tab on the left and the New tab on the top on the center panel.

We used the first method to define our first users. When the registration was complete, we downloaded (using `winscp3`) the user id files (`username.id`) to the client machine and placed them in `C:\Program Files\lotus\notes\data` for easy access at logon time.

Installing the Lotus Notes Client systems on additional Windows machines

Because we wanted to establish a work flow between multiple users and machines, we then repeated the installation of the Lotus Domino Client System on two additional Windows machines.

The steps were the same as before except that we did not have to do the setup again and we did not have to register the users. When the installation completed we downloaded the user id files (`username.id`) to the systems to allow us to log on as those users.

Open source security products

The open source security products we used are Bastille, iptables for both of the firewalls, and Hogwash as the network intrusion prevention system. For details on the implementation of these products, see the following white paper:

<ftp://ftp.software.ibm.com/eserver/zseries/misc/literature/pdf/whitepapers/gm130636.pdf>

Changing the placement of Hogwash

We changed the placement of Hogwash, see “Linux on zSeries network configuration” on page 270, from where it was in the open source security study. Hogwash is an invisible inline packet scrubber, otherwise known as network intrusion prevention system. By invisible, we mean the system that is running Hogwash does not have an IP address, thus it is invisible to the network. And by placing it in front of the first firewall (instead of behind the firewall like in the open source security study), we are able to scrub incoming packets before they reach the firewall – thus preventing certain attacks targeted at the firewall.

iptables

iptables is the standard Linux software firewall. A user can configure a set of iptables rules to log, deny, or permit packets. We currently utilize 2 firewalls in this environment; both are running iptables.

Defining the rules for the firewall between between Public LAN and VLAN674

We defined the rules for the firewall between between Public LAN and VLAN674:

```
# Turn off ICMP redirects
for x in /proc/sys/net/ipv4/conf/*/accept_redirects
do
    echo 0 > $x
done
for x in /proc/sys/net/ipv4/conf/*/send_redirects
do
    echo 0 > $x
done

# Turn on ICMP broadcast rejection
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

# Turn on Source address validation
for x in /proc/sys/net/ipv4/conf/*/rp_filter
do
    echo 1 > $x
done

# Turn on syn_cookies
echo 1 > /proc/sys/net/ipv4/tcp_syncookies

# Flush all the current rules
iptables -F
iptables -F
iptables -F
iptables -F

# Set a policy to drop all forwarded packets
iptables -P FORWARD DROP

#set the forward rules

# allow everybody to get to the nameserver
iptables -A FORWARD -p udp --dport 53 -s 192.168.75.0/24 -j ACCEPT
iptables -A FORWARD -p tcp --dport 53 -s 192.168.75.0/24 -j ACCEPT
```

```

# allow Public lan & VLAN674 to access webSEAL
iptables -A FORWARD -p tcp --dport https -d 192.168.74.112 -s 192.168.74.0/24 -j ACCEPT
iptables -A FORWARD -p tcp --dport https -d 192.168.74.112 -s 192.168.75.0/24 -j ACCEPT

# allow SAMBA/HUB to ssh to all VLANs
iptables -A FORWARD -p tcp --dport 22 -s 192.168.71.108 -j ACCEPT

# allow Nessus to ssh to all VLANs
iptables -A FORWARD -p tcp --dport 22 -s 192.168.71.112 -j ACCEPT

# allow Public lan access to Domino
iptables -A FORWARD -d 192.168.72.117 -s 192.168.75.0/24 -j ACCEPT

# allow NTP Communications from VLAN674 & 673
iptables -A FORWARD -p udp --dport 123 -d 192.168.71.111 -s 192.168.74.0/24 -j ACCEPT
iptables -A FORWARD -p tcp --dport 123 -d 192.168.71.111 -s 192.168.73.0/24 -j ACCEPT
iptables -A FORWARD -p udp --dport 873 -d 192.168.71.111 -s 192.168.74.0/24 -j ACCEPT
iptables -A FORWARD -p tcp --dport 873 -d 192.168.71.111 -s 192.168.73.0/24 -j ACCEPT

# allow Central log server communication from VLAN674 & 673
iptables -A FORWARD -p udp --dport 514 -d 192.168.71.109 -s 192.168.74.0/24 -j ACCEPT
iptables -A FORWARD -p udp --dport 514 -d 192.168.71.110 -s 192.168.74.0/24 -j ACCEPT
iptables -A FORWARD -p udp --dport 514 -d 192.168.71.109 -s 192.168.73.0/24 -j ACCEPT
iptables -A FORWARD -p udp --dport 514 -d 192.168.71.110 -s 192.168.73.0/24 -j ACCEPT

# allow Public lan to ftp server
iptables -A FORWARD -p tcp --dport 21 -d 192.168.73.113 -s 192.168.75.0/24 -j ACCEPT

# allow established and related communication - both ways.
iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT

# LOG everything that is dropped
iptables -A FORWARD -j LOG --log-prefix "DROPPING " --log-level alert

##### INPUT INPUT INPUT #####

## Prevent anything from getting into the firewall
iptables -P INPUT DROP

# allow SSH on the firewall
iptables -A INPUT -p tcp --dport 22 -s 192.168.71.108 -j ACCEPT
iptables -A INPUT -p tcp --sport 22 -s 192.168.71.108 -j ACCEPT
iptables -A INPUT -p tcp --dport 22 -s 192.168.71.112 -j ACCEPT
iptables -A INPUT -p tcp --sport 22 -s 192.168.71.112 -j ACCEPT

# allow established and related communication - both ways.
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

# LOG everything that is dropped
iptables -A INPUT -j LOG --log-prefix "DROPPING " --log-level alert

#list the rules
iptables -L -n

```

Defining the rules for the firewall between VLAN673 and VLAN672

We defined the rules for the firewall between VLAN673 and VLAN672:

```

# Turn off ICMP redirects
for x in /proc/sys/net/ipv4/conf/*/accept_redirects
do
    echo 0 > $x
done
for x in /proc/sys/net/ipv4/conf/*/send_redirects
do
    echo 0 > $x
done

# Turn on ICMP broadcast rejection

```

```

echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

# Turn on Source address validation
#
for x in /proc/sys/net/ipv4/conf*/rp_filter
do
    echo 1 > $x
done

# Turn on syn_cookies
echo 1 > /proc/sys/net/ipv4/tcp_syncookies

# Flush all the current rules
iptables -F INPUT ACCEPT
iptables -F OUTPUT ACCEPT
iptables -F FORWARD ACCEPT
iptables -F

# Set a policy to drop all forwarded packets
iptables -P FORWARD DROP

#### FORWARD FORWARD FORWARD #####
#set the forward rules

# allow everybody to get to the nameserver
iptables -A FORWARD -p udp --dport 53 -s 192.168.71.0/24 -j ACCEPT
iptables -A FORWARD -p udp --dport 53 -s 192.168.72.0/24 -j ACCEPT
iptables -A FORWARD -p udp --dport 53 -s 192.168.73.0/24 -j ACCEPT
iptables -A FORWARD -p udp --dport 53 -s 192.168.74.0/24 -j ACCEPT
iptables -A FORWARD -p udp --dport 53 -s 192.168.75.0/24 -j ACCEPT
iptables -A FORWARD -p tcp --dport 53 -s 192.168.71.0/24 -j ACCEPT
iptables -A FORWARD -p tcp --dport 53 -s 192.168.72.0/24 -j ACCEPT
iptables -A FORWARD -p tcp --dport 53 -s 192.168.73.0/24 -j ACCEPT
iptables -A FORWARD -p tcp --dport 53 -s 192.168.74.0/24 -j ACCEPT
iptables -A FORWARD -p tcp --dport 53 -s 192.168.75.0/24 -j ACCEPT

# allow VLAN674 to access caching proxy
iptables -A FORWARD -p tcp --dport 80 -d 192.168.73.150 -s 192.168.74.0/24 -j
ACCEPT

# allow SAMBA/HUB to ssh to all VLANs
iptables -A FORWARD -p tcp --dport 22 -s 192.168.71.108 -j ACCEPT

# allow Nessus to ssh to all VLANs
iptables -A FORWARD -p tcp --dport 22 -s 192.168.71.112 -j ACCEPT

# allow Public LAN admin access to Domino / WebSeal
iptables -A FORWARD -p tcp -d 192.168.72.117 -s 192.168.75.0/24 -j ACCEPT
iptables -A FORWARD -p tcp --dport 443 -d 192.168.74.112 -s 192.168.75.0/24 -j
ACCEPT

# allow WebSEAL communications from VLAN674 to the Policy Server
iptables -A FORWARD -p tcp --dport 7135 -d 192.168.71.120 -s 192.168.74.112 -j
ACCEPT

# allow WebSEAL communications from VLAN674 to the Load Balancer
iptables -A FORWARD -p tcp --dport 443 -d 192.168.71.98 -s 192.168.74.112 -j
ACCEPT

# allow WebSEAL communications from VLAN674 to LDAP on zLinux
iptables -A FORWARD -p tcp --dport 636 -d 192.168.71.25 -s 192.168.74.112 -j
ACCEPT
iptables -A FORWARD -p tcp --dport 389 -d 192.168.71.25 -s 192.168.74.112 -j
ACCEPT

```

```

# allow Cacing Proxy communications to the Load Balancer
iptables -A FORWARD -p tcp --dport 443 -d 192.168.71.98 -s 192.168.73.150 -j
ACCEPT

# allow NTP Communications from VLAN674 & 673
iptables -A FORWARD -p udp --dport 123 -d 192.168.71.111 -s 192.168.74.0/24 -j
ACCEPT
iptables -A FORWARD -p tcp --dport 123 -d 192.168.71.111 -s 192.168.73.0/24 -j
ACCEPT
iptables -A FORWARD -p udp --dport 873 -d 192.168.71.111 -s 192.168.74.0/24 -j
ACCEPT
iptables -A FORWARD -p tcp --dport 873 -d 192.168.71.111 -s 192.168.73.0/24 -j
ACCEPT

# allow Central log server communication from VLAN674 & 673
iptables -A FORWARD -p udp --dport 514 -d 192.168.71.109 -s 192.168.74.0/24 -j
ACCEPT
iptables -A FORWARD -p udp --dport 514 -d 192.168.71.110 -s 192.168.74.0/24 -j
ACCEPT
iptables -A FORWARD -p udp --dport 514 -d 192.168.71.109 -s 192.168.73.0/24 -j
ACCEPT
iptables -A FORWARD -p udp --dport 514 -d 192.168.71.110 -s 192.168.73.0/24 -j
ACCEPT

# allow Public lan & VLAN674 to ftp server
iptables -A FORWARD -p tcp --dport 21 -d 192.168.73.113 -s 192.168.75.0/24 -j
ACCEPT
iptables -A FORWARD -p tcp --dport 21 -d 192.168.73.113 -s 192.168.74.0/24 -j
ACCEPT

# allow samba to get to webseal
iptables -A FORWARD -p tcp --dport https -d 192.168.74.112 -s 192.168.71.108 -j
ACCEPT

# allow VLAN674 & 673 limited time window to TSM Server for Backups
iptables -A INPUT -p tcp --dport 1500 -d 192.168.71.121 -s 192.168.74.0/24
-j ACCEPT
iptables -A INPUT -p tcp --dport 1500 -d 192.168.71.121 -s 192.168.73.0/24
-j ACCEPT

# allow established and related communication - both ways.
iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT

# LOG everything that is dropped
iptables -A FORWARD -j LOG --log-prefix "DROPPING " --log-level alert

##### INPUT INPUT INPUT #####

## Prevent anything from getting into the firewall
iptables -P INPUT DROP

# allow SSH on the firewall
iptables -A INPUT -p tcp --dport 22 -s 192.168.71.108 -j ACCEPT
iptables -A INPUT -p tcp --sport 22 -s 192.168.71.108 -j ACCEPT
iptables -A INPUT -p tcp --dport 22 -s 192.168.71.112 -j ACCEPT
iptables -A INPUT -p tcp --sport 22 -s 192.168.71.112 -j ACCEPT

# allow established and related communication - both ways.
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

# LOG everything that is dropped

```

```
iptables -A INPUT -j LOG --log-prefix "DROPPING " --log-level alert
#list the rules
iptables -L -n
```

For clearing our firewall rules for testing purposes, we created another script that flushes all the firewall rules in addition to turning the Linux switches back to default. The following is our script:

```
# Flush all the current rules
#
iptables -F
iptables -F
iptables -F
iptables -F
#
# Turn on ICMP redirects
for x in /proc/sys/net/ipv4/conf/*/accept_redirects
do
echo 1 > $x
done

for x in /proc/sys/net/ipv4/conf/*/send_redirects
do
echo 1 > $x
done
#
# Turn off ICMP broadcast rejection
#
echo 0 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
#
# Turn off Source address validation
#
for x in /proc/sys/net/ipv4/conf/*/rp_filter
do
echo 0 > $x
done
#
# Turn off syn_cookies
#
echo 0 > /proc/sys/net/ipv4/tcp_syncookies
#
#list the rules

iptables -L
```

IBM security products

Following are our experiences with the IBM security products we used.

Planning and installing Tivoli Risk Manager (TRM) Host IDs

In order to use Tivoli Risk Manager for Host Intrusion Detection, we first needed to plan it out. We wanted to use the latest version of TRM, which at the time of the test was TRM V4.2. We determined that TRM was supported on SUSE LINUX Enterprise Server 8 SP3. If you are thinking of using TRM on a specific Service Pack, first check with Tivoli support to ensure it is supported. With this information, we went ahead and planned our TRM topology.

We referenced the RedBook “Centralized Risk Management Using Tivoli Risk Manager 4.2” (SG24-6095-00) and followed the recommendation to keep various components of TRM on separate Linux on zSeries images. We had a TRM Event Server in the internal zone (VLAN671), as the central server that collects all the incidents and events and passes them onto the archive database or Tivoli

Enterprise Console® (TEC). We had a TRM Distributed Correlation Server installed in the internal zone to collect events from TRM clients and correlate them to the Event Server. The TRM Event Server required a database, so we installed DB2 V7.2 FP8 (a TRM supported version of DB2) server on a separate system.

Components of our TRM installation

The installation order is very important in setting up TRM. Because there are multiple components involved, you need to take care in planning them out. We installed them in the following order (each component on its own Linux on zSeries image):

1. **TRM database server:** Needed for TEC's event database and TRM's archive database. TEC and TRM configurations require database server information. The database server we are using is DB2 V7.2 FP8.
2. **TRM Event Server:** The Event Server is the last point in the overall layout where sensor events can be independently siphoned off to the archive database. It needs to be installed before the TRM Distributed Correlation Server because DCS requires TRM information during its installation so it knows who to send incidents to.
3. **TRM Distributed Correlation Server:** The DCS collects events from TRM clients and correlates them into incidents before sending incidents to the TRM Event Server. It needs to be installed before TRM clients because TRM clients require DCS information during their installations so they know who to send events to.
4. **TRM Client:** Finally, the endpoint that collects events gets installed. The endpoint needs TRM client to be installed first, and then you can install various adapters and/or configure the TRM client to collect the type of events you are looking for. In our case we installed the Linux Host IDS adapter to collect local host events.

Installing DB2 V7.2 FP8 Server: According the TRM Release Notes, the only DB2 version supported is V7.2 so we decided to go with that. First install the base DB2 V7.2. Then apply FP8. After applying FP8, run `db2iupdt db2inst1`.

Note: The DB2 V7.x product is no longer supported. If you are thinking about installing TRM in your environment, consider using one of the other TRM-supported databases (see the TRM 4.2 Release Notes).

Installing the TRM Event Server: The Event Server requires that you install the following components:

1. Java Runtime Environment (JRE), Version 1.3.1

Tivoli Risk Manager ships with IBMJava2-JRE-1.3.1-5.0, which was what we used for the Event Server. If you encounter problems with the TRM installation wizard, for example, if it doesn't load, then your pre-existing JRE might not be compatible. Use the one that ships with TRM.

2. Tivoli Management Framework, Version 4.1

Note here that the GUI version of the TMF install is not supported on Linux on zSeries, you must install from command line. Follow the *TMF Installation Guide* when installing from the command line. We used the following command to install TMF after generating the scripts with the `WPREINST.SH` script that came with the installation package (see the *TMF Installation Guide* for more details on using the `WPREINST.SH` script):

```
littec:/usr/local/Tivoli/installdir # ./wserver -c /opt/TMFinstall
BIN=/opt/Tivoli/TMF/bin LIB=/opt/Tivoli/TMF/lib ALIDB=/opt/Tivoli/TMF/database
MAN=/opt/Tivoli/TMF/man APPD=/opt/Tivoli/TMF/X11 CAT=/opt/Tivoli/TMF/cat
RN=NoonTide-Region AutoStart=1 SetPort=1 CreatePaths=1 IP=Tivoli4Ever
```

Then you must install the TMF license with the `odadmin set_platform_license` license command. See the *TMF Command Reference* for how to use this command.

Set the TMF environment by running:

```
. /etc/Tivoli/setup_env.sh
```

Install TMF patches 10E, 13, 14 and 15.

Extract the patches in some temporary directory, for example;

```
/opt/TMFinstall/fixpacks:
```

```
tar -xzvf 4.1-TMF-0013.tar
tar -xzvf 4.1-TMF-0014.tar
tar -xzvf 4.1-TMF-0015.tar
tar -xzvf 4.1-TMF-0010E.tar
```

Now install the patches:

```
# cd /opt/TMFinstall/fixpacks
# wpatch -c /opt/TMFinstall/fixpacks/41TMF010E -i 41TMF010E LCF_NEW=! LCF41=!
# wpatch -c /opt/TMFinstall/fixpacks/41TMF014 -i 41TMF014 LCF_NEW=! LCF41=!
#wpatch -c /opt/TMFinstall/fixpacks/41TMF015 -i 41TMF015 LCF_NEW=! LCF41=!
# wpatch -c /opt/TMFinstall/fixpacks/41TMF013 -i 41TMF013 LCF_NEW=! LCF41=!
```

Create an administrator with the `wcrtdadmin` command. See the *TMF Command Reference* for details on command options.

3. Tivoli Enterprise Console, Version 3.9

Install DB2 client, it must be the same version as the server, in our case V7.2 FP8. Catalogue a local DB2 node that communicates with the remote DB2 server:

```
db2 catalog tcpip node litrddb remote litrddb server TEC
```

Where TEC is the DB2 server port number, in our case, it was set to port 3700.

Catalog a local database:

```
db2 catalog database TECDB as TECDB at node litrddb
```

Then create the TEC RIM component (this is the event database connector).

This is the command we ran (See the *TMF Command Reference* for more details on the parameters):

```
wcrtrim -v DB2 -h littec -d TECDB -u db2inst1 -H /usr/IBMd2/V7.1 -s
tcpip -I /home/db2inst1 -t db2inst1 TECRIM
RDBMS password: db2inst1
```

On SUSE LINUX Enterprise Server 8 SP3, there is a known problem with the TEC installation. TEC requires the `compress` utility, and more recent version of Linux distributions from SUSE LINUX do not include the `compress` utility. Before you progress with the TEC installation, follow our workaround in “Problems we encountered” on page 294.

Use the GUI Installation Wizard to install TEC. First select “Configure the database”, make sure to select “DB2 client” for the RIM host parameter. After the database is configured, go back to the main menu and select “Install the TEC components” and then select the following components to install: Event server, User interface server, Event console, and Adapter Configuration Facility. You can also install these components with the command `winstall` (See *TMF Command Reference* and *TEC Installation Guide* for more details).

4. Tivoli Risk Manager, Version 4.2, Event Server configuration

Start the GUI install with the following command:

```
littec:/opt/TRMinstall# java -Dis.javahome=/opt/IBMJava2-s390-131/jre \
cp ./riskmgr42.jar run
```

Follow through the GUI panels and make sure to select the “Event Server” configuration. In the database configuration panel, point to the right JDBC driver path. In our case, this was /usr/IBMDB2/V7.1/java/db2java.zip.

5. Starting the TRM Event Server.

First make sure TMF is running by issuing the `ps -ef` command and looking for an `oserv` process. Then make sure the TEC event server is started by running `wstatesvr`; if it’s not running, start it with `wstartesvr`. Setup the TRM environment variables by executing the script `/etc/Tivoli/rma_eif_env.sh`. And now start TRM by running `rmagent &`.

Verify that it is running by executing `wrmadmin -info`, and you should see the following:

```
littec:/usr/IBMDB2/V7.1/instance # wrmadmin -info
HRMRM0003I Tivoli Risk Manager Version 4.2.
HRMRM0004I The Tivoli Risk Manager Agent is active.

      Tivoli Risk Manager Agent Component Status
      =====
      Engines
          correlation: Running

      Event sources
          eif_receiver: Running

      Event destinations
          incident_sender: Running
          db_sender: Running
```

Setting up the TRM Distributed Correlation Server: The following are the steps in setting up the TRM Distributed Correlation Server:

1. Install the DB2 client, and catalogue a local node to point to the DB2 server.
2. Catalogue a local database that connects to the DB2 server’s TRM archive table. In our case, the database name is TECDB. See 3 on page 292 for the specific DB2 commands to do these tasks.
3. Using the TRM installation GUI as above, select to install the DCS configuration. During configuration, point to the TRM Event Server for sending incidents. Start it the same way as you would the Event Server.

Setting up the TRM Client & Host IDS Adapter: The following are the steps in setting up the TRM Client & Host IDS Adapter:

1. Install JRE 1.3.1.
2. Install the TRM client configuration using the TRM installation GUI.
3. Use the EventMonitor’s `launch.sh` to install the Host IDS Adapter. You can download the Host IDS Adapter and the EventMonitor package for Linux from the TRM support website. During configuration of the client, point to the DCS server for sending events. Start it the same way as you would the Event Server or DCS.

Checking for incidents

Crystal Report is not supported on Linux on zSeries so you need to install Crystal Report which is a TRM install option on the Windows platform only, and configure it point to your archive database. TRM provides 22 packaged reports which can be parameterized at execution time to provide a wide variety of views into the Risk Manager event/archive database.

At the time of the security testing, we didn’t have time to secure a Windows machine to install Crystal Report on so we used the `wtdump` command to view

incidents reported. The command dumps the whole database that contains all the incidents. An incident entry looks similar to this:

```
1~3839~65537~1116304713(May 17 00:38:33 2005)
### EVENT ###
RM_SrcDstCat_Incident;rm_WindowSize=600000;repeat_count=4;rm_CategoryDisplay
Names=['Authentication denied'];msg='Category: SECAUTH.DENY. Suspicious activity
at litxwas03.';sub_source=INCIDENT;
rm_FirstEventTime=1115648843;rm_Level=12.0;rm_CustomerID=N/A;
rm_ThresholdLevel=10.0;rm_EventCount=4;hostname='SECAUTH.DENY :
litxwas03 => litxwas03';
rm_CategoryTokens=['SECAUTH.DENY'];rm_Stamp='Tue May 17 00:38:32 2005';
rm_DestinationTokens=['litxwas03'];
source=RISKMgr;severity=CRITICAL;rm_AgentNormalized=true;
rm_Sensors=['OS_Linux/litxwas03'];rm_Stamp32=1116304712;
rm_LastEventTime=1115648843;
rm_Signatures=['N/A'];rm_SourceTokens=['litxwas03'];END

### END EVENT ###
```

Problems we encountered

Following are the problems we encountered in planning and installing Tivoli Risk Manager (TRM) Host IDs:

1. SUSE LINUX Enterprise Server 8 SP3 compress, uncompress issue:

On SUSE LINUX Enterprise Server 8 SP3, the compress, uncompress commands point to a dummy shell script and the gzip command. Tivoli Management Framework's wbkupdb command calls compress/uncompress and does not work right unless it is the actual command. In our case, we had an indefinite hang. Installing TEC through the GUI installer backs up the objects first so that hung as well.

This is a known problem and it is documented in TEC's Release Notes, <http://publib.boulder.ibm.com/infocenter/tiv3help/index.jsp?topic=/com.ibm.itecrn.doc/econmst20.htm>

This documentation states "For SUSE and SUSE LINUX Enterprise Server (SLES) distributions of Linux, if the compress utility is not installed, you might experience problems, such as the rule base not being loaded or the event server not starting due to a missing rule base. The Tivoli Enterprise Console product requires the compress utility, and more recent versions of Linux distributions from SUSE do not include the compress utility.

Workaround: You could obtain the compress utility from an older level of Linux distributions from SUSE."

Instead of copying the compress binary from an older system, we built the utility from source and found that it links the binaries for you and worked just as well. The following are the steps for building the ncompress utility:

- a. Get the source package from
`ftp://ftp.nectec.or.th/pub/linux-distributions/Debian/pool/non-free/n/ncompress/ncompress_4.2.4.orig.tar.gz`
- b. Unpack it:
`tar xvzf ncompress_4.2.4.orig.tar.gz`
- c. Run the build script:
`littec:~/ncompress/ncompress-4.2.4.orig # ./build`
- d. Select "c" for compile
- e. Select "i" for install
- f. Test it with TMF's wbkupdb command (it should complete relatively quickly):
`littec:~/ # wbkupdb`

Starting the snapshot of the database files for littec...

.....
.....

Backup Complete.

2. DB2 server and client must be the same version.

If you have a remote DB2 server as we do, and installed DB2 clients on your TRM Event Server and DCS and configured them to send events and incidents to the DB2 server, you must have the same versions on both your server and client. At the time of this testing, the only DB2 version supported on Linux by TRM was DB2 V7.2 FP8, so we had to make sure that all the clients had this version as well. Otherwise, you will see a `db_sender: Failed Retrying` status when you do a `wrmadmin -info` to check on your agent status.

3. Incident sender is failing.

Another problem we ran into was the `incident_sender` component on the TRM Event Server getting a `Failed Retrying` status during a `wrmadmin -info` command. It turned out that we had the wrong login name in `$/RMADHOME/etc/incident_sender.conf`. To check what our login name was, we did a `wgetadmin` command, and our login name is indicated below in bold. (Note that the login name is just the front part before the `@` sign):

```
littec:~ # wgetadmin
Administrator: Root_NoonTide-Region
logins: root@littec.ltic.pok.ibm.com
roles: global super, senior, admin, user, install_client, install_product,
policy, RIM_view, RIM_update, Query_execute, Query_view, Query_edit,
ACF_glopol, ACF_polmod, ACF_rwdist, ACF_readonly
security_group_any_admin user
Root_NoonTide-Region admin, user, rconnect
TecUIServer super, senior, admin, user, install_client,
install_product, policy, RIM_view, RIM_update, Query_execute, Query_view,
Query_edit, ACF_glopol, ACF_polmod, ACF_rwdist, ACF_readonly
EventServer super, senior, admin, user, install_client,
install_product, policy, RIM_view, RIM_update, Query_execute, Query_view,
Query_edit, ACF_glopol, ACF_polmod, ACF_rwdist, ACF_readonly
ACPdefault super, senior, admin, user, install_client,
install_product, policy, RIM_view, RIM_update, Query_execute, Query_view,
Query_edit, ACF_glopol, ACF_polmod, ACF_rwdist, ACF_readonly
notice groups: TME Administration, TME Authorization, TME Diagnostics, TME Scheduler
```

Now you want to make sure that everything is working. Restart the `rmagent` with `wrmadmin -r`, wait a few minutes, and then run `wrmadmin -info`.

```
littec:/opt/RISKMG/et # wrmadmin -info
HRMRM0003I Tivoli Risk Manager Version 4.2.
HRMRM0004I The Tivoli Risk Manager Agent is active.
```

```
Tivoli Risk Manager Agent Component Status
=====
Engines
    correlation: Running

Event sources
    eif_receiver: Running

Event destinations
    incident_sender: Running
    db_sender: Running
```

Authenticating and authorizing Web transactions using Tivoli Access Manager and TAM WebSeal

Tivoli Access Manager V5.1 and Tivoli Access Manager WebSeal were used in our environment to authenticate and authorize Web transactions. We connected TAM to a backend z/OS LDAP server.

First run the `ivrgy_tool` command that comes with TAM to setup TAM schema in the LDAP server:

```
littam71:/opt/PolicyDirector/sbin # ./ivrgy_tool -h z0eip -p 3389 -D
"cn=webadm" -w webadm schema
```

We wanted to use SSL connection between LDAP and TAM/WebSeal. So we created a key database with a user or server certificate with 1024-bit RSA key, using the `gskkyman` tool in z/OS USS. In the `/Z0/etc/ldap/slapd.conf` file on z/OS, we specified the port for SSL connection as well as the key database information (Z0 is the name of our z/OS system):

```
9 listen ldaps://:6636
10
11 sslAuth serverAuth
12 sslKeyRingFile /Z0/etc/ldap/J80LDAP/keys/Z0ldaptim.kdb
13 sslCertificate cert_ldap
14 sslKeyRingFilePW password
15 sslCipherSpecs ANY
16 sslKeyRingPWStashFile /Z0/etc/ldap/J80LDAP/keys/Z0ldaptim.sth
```

We transferred the `Z0ldaptim.kdb` key database over to the TAM and WebSeal systems. Now we wanted to configure TAM and WebSeal as usual. When configuring the TAM runtime, we chose the regular non-SSL LDAP port, but when configuring the policy server and WebSeal, we chose the SSL LDAP port, making sure to point to the key database that was just transferred.

As noted in “Setting SSL tunneling on in the WebSphere Application Server Edge Component Caching Proxy V5.1” on page 279, we used WebSeal to create a SSL proxy junction that tunneled authorized traffic through to the WebSphere Application Severcluster. To create a SSL proxy junction we ran the following command from the TAM `pdadmin` command prompt:

Problems we encountered

If you have any other suffix defined in z/OS LDAP other than:

```
34 suffix "o=ibm, c=us"
35 suffix "secAuthority=Default"
```

Then you need to tell TAM to ignore the other suffixes defined in `slapd.conf` on your z/OS LDAP. Do this by editing `/opt/PolicyDirector/etc/ldap.conf` with:

```
ignore-suffix = sysplex=utcp1xj8,o=ibm,c=us
```

The suffix `sysplex=utcp1xj8,o=ibm,c=us` was a complete suffix (different from the TAM accepted `o=ibm,c=us`) that was used for a z/OS RACF backend.

Authenticating Linux users using RACF and LDAP on z/OS

The guide for which we used as a reference is found at the following link:

<http://www.ibm.com/developerworks/eserver/library/es-sles-ldap/>

The above documentation is geared toward SUSE LINUX Enterprise Server 9 setup. Our systems are running SUSE LINUX Enterprise Server 8 SP3 31bit. There are differences between the changes required in `/etc/pam.d/sshd`. Here is our `/etc/pam.d/sshd` after we modified it:

```
litzweb:/etc/pam.d # cat sshd
#%PAM-1.0
auth requisite      pam_nologin.so
auth required      pam_nologin.so
auth required      pam_env.so
#authentication against an LDAP server
auth sufficient    pam_ldap.so
```

```

auth required      pam_unix.so use_first_pass
auth      required pam_unix2.so use_first_pass
#LDAP Server account and session
account sufficient pam_ldap.so
account required  pam_unix2.so
account required  pam_nologin.so
session required  pam_unix2.so    none # trace or debug
session required  pam_limits.so
password sufficient pam_ldap.so
password required  pam_unix2.so  use_first_pass use_authtok

```

The other addition we had to make was in the `/etc/openldap/ldap.conf` file to include the non-unique port (3389) we use for LDAP on z/OS.

```

litzweb:/etc/openldap # cat ldap.conf
# $OpenLDAP: pkg/ldap/libraries/libldap/ldap.conf,v 1.9 2000/09/04 19:57:01 kurt Exp $
#
# LDAP Defaults
#
# See ldap.conf(5) for details
# This file should be world readable but not world writable.
host z0eip.pdl.pok.ibm.com
port 3389
base sysplex=UTCPLXJ8,o=IBM,c=US
binddn racfid=WEBADM,profiletype=user,sysplex=UTCPLXJ8,o=IBM,c=US
bindpw webadm
ldap_version 3
pam_login_attribute racfid

#BASE dc=example, dc=com
#URI ldap://ldap.example.com ldap://ldap-master.example.com:666

#SIZELIMIT      12
#TIMELIMIT      15
#DEREF          never

```

The rest of the guide is accurate for both SUSE LINUX Enterprise Server 8 and SUSE LINUX Enterprise Server 9.

Independent service vendor (ISV) security products

Following are some of the ISV security products that we used and tested.

Installing TrendMicro's ScanMail

In order to protect the mail users from viruses, we decided to install the TrendMicro ScanMail program. We used V2.6 with SUSE SLES 8 SP3.

We downloaded the product with the following three files:

1. `readme-sm1n-zlinux26-b1529.txt` - basic description and install instructions
2. `sm1n26-gsg.pdf` - installation and users guide
3. `sm1n26-sp1-zlinux-b1529.tar` - program iteslf

We stopped the Domino server by issuing "quit" from the start up session.

We untarred the program file and began the installation by issuing the `./sminst` command and were surprised to find it did not execute. We checked the mode of the file and found it was not executable and so changed the mode to 555 for this execution. (`chmod 555 sminst`).

The installation went very easily with only a few questions to answer. We had to identify the Domino administrator, where the Domino Data Directory was, and a few other things. The process took only a few minutes.

| On completion we restarted the Domino server and went to the Windows machine
| to try and send some mail. Everything went well, but the question remained: "Are
| we checking for viruses?". To answer that question, Trend Micro provided an inert
| virus test file to try and pass in the mail.

| **Testing to see if it detects viruses**

| In order to test if the system is detecting viruses, you need to introduce a virus into
| the system. From Trend Micro's Getting Started Guide:

| "The European Institute for Computer Antivirus Research, or EICAR has developed
| a test script that can be used to test your antivirus software. This script is an inert
| text file whose binary pattern is included in the virus pattern file from most antivirus
| vendors. *It is not a virus and does not contain any program code.*"

| We tried to create a file on the Windows system containing the test virus but failed.
| Each time we tried to file the text, the Windows virus scan program (Symantic's
| Norton Antivirus program) isolated the file and destroyed it.

| In order to make the test, we had to stop the antivirus program temporarily. To do
| this we had to go into the Systems settings and disable it.

| Finally we were able to create the test file by opening a Notepad file and inserting
| the following string

| X50!P%@AP[4\PZX54(P^)7CC)7}\$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!\$H+H*

| and filing it under the name eicar.com.

| We then logged on to a Lotus Notes client and sent a message with the eicar.com
| file as an attachment. The file was caught and isolated. The receiver got a message
| with the attachment, but the attachment contained only one character of data and a
| "RED" warning message that the file had a virus. The sender was not notified of the
| problem.

| We went back to the "ScanMail Getting Started Guide" and found that we missed
| setting some ScanMail options. Following the instructions on page 4-14, we logged
| on to Lotus Notes as the Administrator, opened the ScanMail data base, and went
| to the section on "Virus Notification". There we turned on the options to notify the
| sender, the receiver, and the Administrator of any virus detected. We retested the
| virus detection and found that in addition to the "RED note" the sender received,
| the sender, receiver, and Administrator, each received an additional note identifying
| the problem and its source.

| **Installing TrendMicro's ServerProtect**

| Trend Micro ServerProtect™ for Linux™ provides comprehensive protection against
| computer viruses, Trojans, and worms for file servers based on the Linux operating
| system. Managed through an intuitive, portable Web-based console or Linux
| command line console, ServerProtect provides centralized virus scanning, pattern
| updates, event reporting and antivirus configuration.

| TrendMicro ServerProtect is only supported on SUSE (SLES 8), and not Red Hat
| (RHEL 3). We tried installing ServerProtect on RHEL 3, but it failed with an
| unsupported kernel version when installing the rpm.

|
| Instead, we installed Server Protect on SLES 8 SP3 31bit. Installation of the
| product is straightforward; run the SProtectLinux-1.3.0.SLES_S390.s390.bin install
| EXEC from the install source.

|
| To access the ServerProtect Web Console, access it through the URL at port
| *http://xxx:14942/* or *https://xxx:14943/*.

|
| Trend Micro recommends testing ServerProtect and confirming that it works by
| using the EICAR test file, which is a safe way to confirm that your antivirus software
| is properly installed and configured.

|
| To test the ServerProtect installation with EICAR:

- | 1. Open an ASCII text file and copy the following 68-character string to it:
| X50!P%@AP[4\PZX54(P^)7CC)7}\$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!\$H+H*
- | 2. Save the file as eicar_test.com to a temp directory and then close it.
- | 3. Access ServerProtect Web Console at *http://192.168.71.119:14942/*. By default,
| it will have /root in its list of directories to scan; specify / and add it to list to
| scan entire system.
- | 4. Click *save and scan*. ServerProtect locates the fake virus as displayed in
| Figure 51 on page 300:
|

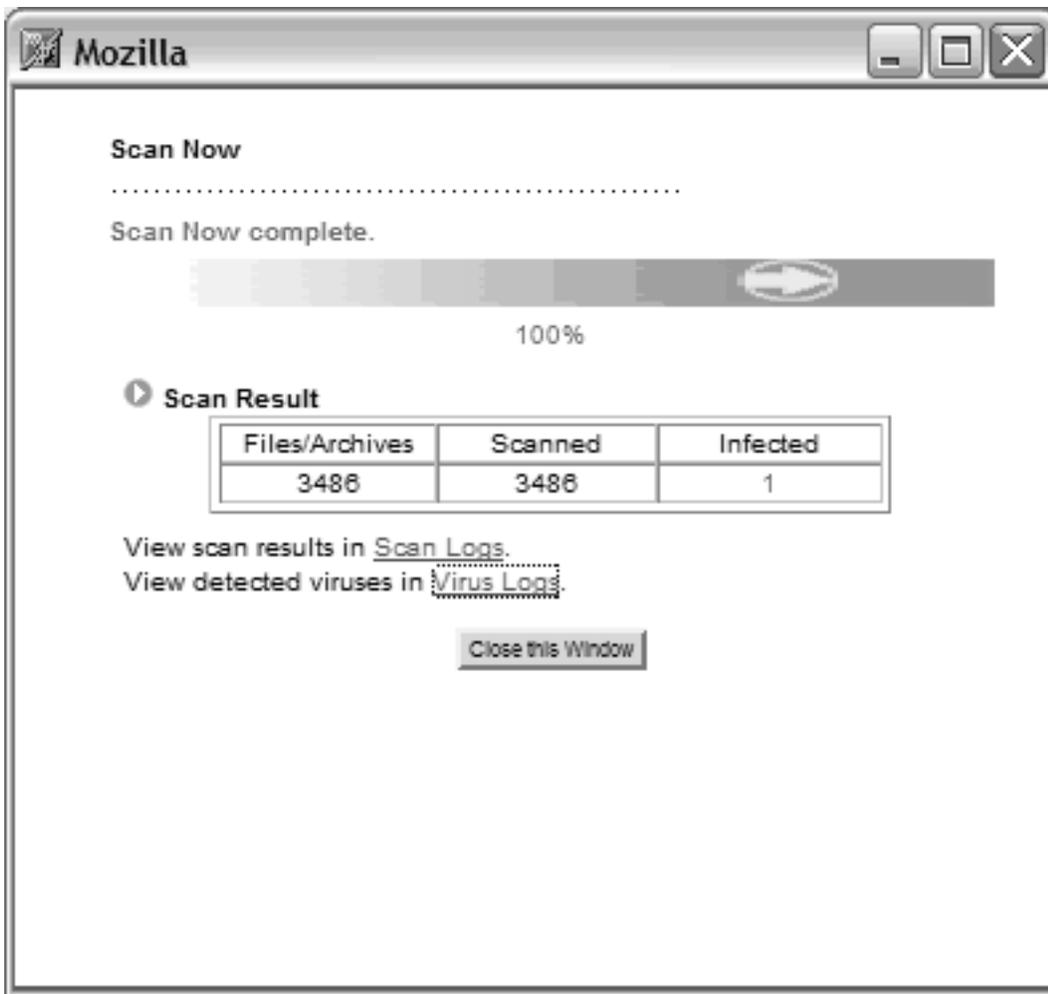


Figure 51. ServerProtect's Scan Complete display.

5. Click *Virus Logs* that shows details about the virus and how it was dealt with. See Figure 52 on page 301.

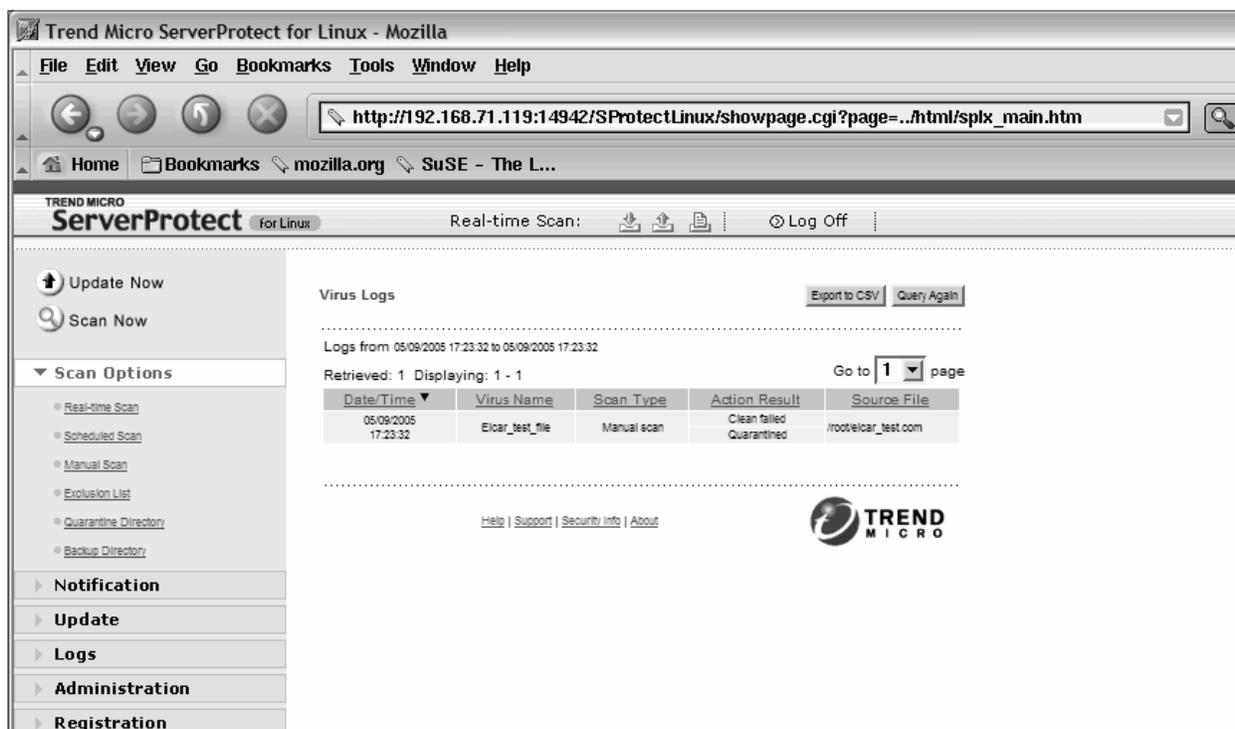


Figure 52. ServerProtect's "Virus Logs" display.

In our case, the virus was quarantined and moved to `/opt/TrendMicro/SProtectLinux/SPLX.Quarantine` where it can be deleted by the user.

Security testing

We ran nmap scans on both our firewalls to make sure that only ports and IP addresses that are open through the firewall are accessible. For more details on running nmap, see the open source security paper at:

<ftp://ftp.software.ibm.com/eserver/zseries/misc/literature/pdf/whitepapers/gm130636.pdf>

We brought in an internal ethical hacker who worked with us during the open source security study. This time, the hacker was able to bring down our VM image with attacks against one of our Linux servers. We took the dump and analyzed it to find that the problem was reported in APAR VM63634 . We will need to apply the fix and re-test to see that it works, so apply the fix to avoid the problem.

Hogwash filtered many attacks with its base `stock.rules` and on the Hogwash console we would see messages alerting that packets were dropped. Tivoli Risk Manager reported many alerts as well. You can check Tivoli Risk Manager incidents by using Crystal Report or with the `wtdump1` command and redirect the output to a file that you can analyze. For details on the command and Crystal Report, see "Planning and installing Tivoli Risk Manager (TRM) Host IDs" on page 290.

Security: Next steps

The security test is an on-going project; we will improve our security environment by experimenting with new products and installing the latest security patches.

Chapter 21. Future Linux on zSeries projects

Following are some areas of future testing for the Linux on zSeries team.

Migration

The Integration Test team is going to migrate or transition the security environment from the 2.4 kernel to the 2.6 kernel, following the transition guide which you can find at:

<http://www.ibm.com/servers/eserver/zseries/library/techpapers/pdf/linux-14mg.pdf>

High availability

The team is also planning on creating a highly available Linux server environment, with interoperability with z/OS and taking advantage of high availability features offered in middleware products as well as on each OS.

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 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/websphere/httpservers/library.html>
- IBM TechDocs (flashes, white papers, etc.), available at www.ibm.com/support/techdocs/
- IBM WebSphere Application Server for z/OS and OS/390 documentation, available at http://www.ibm.com/software/webservers/appserv/zos_os390/library/
- IBM WebSphere Studio documentation, available at <http://www.ibm.com/software/websphere/studio/library.html>
- IBM WebSphere Studio Workload Simulator documentation, available at www.ibm.com/software/awdtools/studioworkloadsimulator/library/
- *Java Servlet Specification, v2.2*, available at <http://java.sun.com/products/servlet/>
- *Java 2 Platform Enterprise Edition Specification, v1.2*, available at <http://java.sun.com/products/j2ee/>
- *JavaServer Pages Specification, Version 1.1*, available at <http://java.sun.com/products/jsp/>
- *J2EE Connector Architecture*, available at <http://java.sun.com/j2ee/connector/>
- Tivoli Risk Manager product manuals, available at <http://publib.boulder.ibm.com/tividd/td/RiskManager4.2.html>
- Tivoli Access Manager Info Center, available at: http://publib.boulder.ibm.com/infocenter/tiv2help/index.jsp?toc=/com.ibm.itame.doc_5.1/toc.xml
- Exploring Open Source Security for a Linux Server Environment available at <ftp://ftp.software.ibm.com/eserver/zseries/misc/literature/pdf/whitepapers/gm130636.pdf>
- DB2 Info Center located at <http://publib.boulder.ibm.com/infocenter/db2help/index.jsp?topic=/com.ibm.db2.udb.doc/ad/c0006975.htm>
- IBM WebSphere Application Server Network Deployment Edge Components Info Center located at

| <http://www-306.ibm.com/software/webservers/appserv/doc/v51/ec/infocenter/index.html>

- IBM WebSphere Application Server Version 5.1.x Info Center located at:

| <http://publib.boulder.ibm.com/infocenter/wasinfo/v5r1/index.jsp>

Appendix A. Some of our parmlib members

This section describes how we have set up some of our parmlib members for z/OS. Table 17 summarizes our new and changed parmlib members for z/OS V1R5 and z/OS.e V1R5. Samples of some of our parmlib members are available on the Samples page of our Web site.

Table 17. Summary of our parmlib changes for z/OS V1R5 and z/OS.e V1R5

Member name	z/OS release	Change summary	Related to
IFAPRDxx	z/OS.e V1R5	No changes from z/OS.e V1R3. (See our December 2002 edition.)	z/OS.e, dynamic enablement
IEASYMPT	z/OS.e V1R5	No changes from z/OS.e V1R3. (See our December 2002 edition.)	z/OS.e, dynamic enablement
IEASYSxx	z/OS.e V1R5	No changes from z/OS.e V1R3. (See our December 2002 edition.)	z/OS.e
LOADxx	z/OS V1R5	Changed the PARMLIB statement to use SYS1.PETR15.PARMLIB	concatenated parmlib
(Actually in SYS0.IPLPARM. See note below.)	z/OS.e V1R5	No changes from z/OS.e V1R3. (See our December 2002 edition.)	z/OS.e
LPALSTxx	z/OS V1R5	Added: SYS1.SIATLPA	JES3
PROGxx (APF additions)	z/OS V1R5	Added: SYS1.SHASLINK SYS1.SHASMIG	JES2
PROGyy (LNKLST)	z/OS V1R5	Added to LNKLSTxx: SYS1.SHASLINK SYS1.SHASMIG	JES2
		Added to LNKLSTxx: SYS1.SIATLIB SYS1.SIATLINK SYS1.SIATMIG	JES3

Note: As of our OS/390 R6 testing, we changed LOADxx to use a generic name, IEASYMPT, for our IEASYMxx member. We have successfully used the name IEASYMPT for our migrations through all subsequent releases of OS/390 and z/OS. Only the entries in SYS0.IPLPARM changed.

Parmlib members

Appendix B. Some of our RMF reports

In this appendix we include some of our RMF reports, as indicated in “z/OS performance” on page 52.

RMF Monitor I post processor summary report

The following figure 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. This report contains information for the same date and time interval as the report in Figure 55 on page 309.

R M F S U M M A R Y R E P O R T																	
OS/390		SYSTEM ID JA0						START 01/18/2001-10.00.00		INTERVAL 00.30.00							
REL. 02.10.00		RPT VERSION 02.10.00						END 01/18/2001-10.30.00		CYCLE 0.100 SECONDS							
INT	CPU	DASD	DASD	TAPE	JOB	JOB	TSO	TSO	STC	STC	ASCH	ASCH	OMVS	OMVS	SWAP	DEMAND	
MM.SS	BUSY	RESP	RATE	RATE	MAX	AVE	MAX	AVE	MAX	AVE	MAX	AVE	MAX	AVE	RATE	PAGING	
	OS/390				SYSTEM ID JA0												
30.00	52.0	16	78.6	0.0	0	0	0	0	283	282	0	0	3	3	0.00	0.00	
	OS/390				SYSTEM ID JB0												
30.00	55.9	14	89.9	0.0	113	113	0	0	170	169	0	0	2	2	0.00	0.01	
	OS/390				SYSTEM ID JB0												
30.00	36.5	16	907.7	0.0	1	1	6	6	300	292	0	0	3	3	0.00	0.01	
	OS390				SYSTEM ID JC0												
29.59	25.3	11	151.0	0.0	0	0	0	0	280	279	0	0	1	1	0.00	0.22	
	OS/390				SYSTEM ID JE0												
29.59	26.0	13	142.1	0.0	0	0	0	0	280	278	0	0	3	3	0.00	0.12	
	z/OS V1R1				SYSTEM ID Z2												
30.00	13.3	22	33.1		0	0	0	0	264	263	0	0	4	4	0.00	0.00	
	z/OS V1R1				SYSTEM ID Z3												
30.00	13.8	21	33.5		0	0	0	0	273	271	1	0	7	5	0.00	0.02	
	z/OS V1R1				SYSTEM ID TPN												
29.59	67.8	5	74.5	0.0	0	0	1	1	277	275	0	0	2	2	0.00	0.00	
	z/OS V1R1				SYSTEM ID JG0												
30.00	23.8	11	387.2	0.0	113	113	3	3	175	173	1	0	8	4	0.00	0.00	
	z/OS V1R1				SYSTEM ID JH0												
29.59	30.7	17	610.1	0.0	0	0	2	2	282	280	0	0	2	2	0.00	0.06	
	z/OS V1R1				SYSTEM ID J90												
30.00	25.1	9	641.4	0.0	115	113	6	4	181	176	1	0	13	6	0.00	0.42	
	z/OS V1R1				SYSTEM ID JF0												
29.59	35.9	12	1022	0.0	0	0	0	0	289	287	0	0	2	2	0.00	0.25	
	z/OS V1R1				SYSTEM ID Z0												
30.00	31.5	9	1347	0.0	114	113	10	9	325	312	1	0	42	35	0.00	0.10	

Figure 53. Example RMF Monitor I post processor summary report

RMF Monitor III online sysplex summary report

The following figure contains information from the *RMF Monitor III Online Sysplex Summary Report*. 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 1130 CICS transactions/second. Note that this report is a snapshot, as opposed to Figure 55 on page 309, which is based on a 1/2-hour interval.

RMF reports

```

HARDCOPY RMF 2.10.0 Sysplex Summary - UTCPLXJ8 Line
WLM Samples: 479 Systems: 14 Date: 01/18/01 Time: 10.05.00 Range: 1
>>>>>>>XXXXXXXXXXXXXXXXXXXX<<<<<<<<

Service Definition: WLMDEF01 Installed at: 12/07/00, 09.35
Active Policy: WLMPOL01 Activated at: 12/07/00, 09.50
----- Goals versus Actuals ----- Trans --Avg. Resp
Exec Vel --- Response Time --- Perf Ended WAIT EXECU
Name T I Goal Act ---Goal--- --Actual-- Indx Rate Time Tim
BATCH W 83 0.025 1.076 59.8
BATCHHI S 2 50 83 0.60 0.000
DISCR S D 88 0.025 1.076 59.8
CICS W N/A 1130 0.000 0.08
CICS S 2 N/A 0.600 80% 99% 0.50 961.9 0.000 0.08
CICS CONV S 3 N/A 10.00 50% 51% 1.00 5.883 0.000 13.4
CICS CP S 1 N/A 0.500 90% 100% 0.50 1.325 0.000 0.00
CICS DEFA S 3 N/A 1.000 90% 100% 0.50 1.308 0.000 0.26
CICS MISC S 3 N/A 1.000 90% 100% 0.50 159.8 0.000 0.01
CICS RGN S 2 60 54 1.11 0.000
IMS W N/A 44.15 0.000 0.14
IMSTMHI S 2 N/A 0.500 90% 96% 0.70 42.17 0.000 0.14
IMSTMLOW S 4 N/A 1.000 90% 100% 0.50 1.983 0.000 0.12
STC W 65 0.642 0.002 43.1
DB2HIGH S 2 50 56 0.90 0.000 0.000 0.00
IMS S 2 50 57 0.88 0.000
IMSHIGH S 2 60 50 1.20 0.000
OMVS S 65 0.333 0.002 1.28
1 2 30 75 0.40 0.000 0.000 0.00
2 3 20 50 0.40 0.125 0.002 8.72
3 5 10 69 0.14 0.208 0.002 1.96
OMVSKERN S 1 40 72 0.55 0.308 0.002 6.79
TPNS S 2 70 78 0.89 0.000 0.000 0.00
SYSTEM W 46 0.042 1.201 32.6
SYSOTHER S N/A 63 N/A 0.000 0.000 0.00
SYSSTC S N/A 39 N/A 0.042
SYSTEM S N/A 59 N/A 0.000 0.000 0.00
TSO W 80 1.750 0.025 2.53
TSO S 2 80 2.000 AVG 2.555 AVG 1.28 1.750 0.025 2.53
:

```

Figure 54. Example RMF Monitor III online sysplex summary report

RMF workload activity report in WLM goal mode

The following figure illustrates a couple of sections from our RMF *Workload Activity Report* in goal mode. This report is based on a 1/2-hour interval. Highlighted on the report you see 99.2% of our CICS transactions are completing in 0.5 seconds, and our CICS workload is processing 1130.54 transactions per second.

On/Off Capacity on Demand Testing

The following figure illustrates a couple of sections from our RMF *Workload Activity Report* with our On/Off Capacity OLTP workloads. We converted our 2066 processor from a model 004 to a model A02. With all our standard workloads running we concurrently upgraded from a model A02 to a model 002, then to a model 003 and finally back to a model 004. We observed no disruption to our workloads. We did observe an expected decrease in CP utilization while at model types A02, 002 and 003.

Note: These are random OLTP workloads as this is not a controlled benchmarking environment.

Samples: 120 System: J80 Date: 09/29/04 Time: 08.54.00 Range: 120

Partition: J80 2066 Model A02
 CPC Capacity: 50 Weight % of Max: 40.0 4h MSU Average: 34
 Image Capacity: 50 WLM Capping %: **** 4h MSU Maximum: 52

Partition	MSU Def	MSU Act	Cap Def	Proc Num	Logical Effect	Util % Total	- Physical LPAR	Util % Effect	- Physical Total
*CP							0.3	89.8	90.1
J80	0	51	NO	2.0	85.8	86.0	0.1	89.8	90.0
PHYSICAL							0.2		0.2

Samples: 120 System: J80 Date: 09/29/04 Time: 10.57.00 Range: 120

Partition: J80 2066 Model 002
 CPC Capacity: 60 Weight % of Max: 40.0 4h MSU Average: 38
 Image Capacity: 60 WLM Capping %: **** 4h MSU Maximum: 54

Partition	MSU Def	MSU Act	Cap Def	Proc Num	Logical Effect	Util % Total	- Physical LPAR	Util % Effect	- Physical Total
*CP							0.3	85.8	86.1
J80	0	51	NO	2.0	85.8	86.0	0.1	85.8	86.0
PHYSICAL							0.2		0.2

Graphic version of this report is not available.

Samples: 120 System: J80 Date: 09/29/04 Time: 13.04.00 Range: 120 Sec

Partition: J80 2066 Model 003
 CPC Capacity: 84 Weight % of Max: 40.0 4h MSU Average: 56
 Image Capacity: 84 WLM Capping %: **** 4h MSU Maximum: 75

Partition	MSU Def	MSU Act	Cap Def	Proc Num	Logical Effect	Util % Total	- Physical LPAR	Util % Effect	- Physical Total
*CP							0.1	84.2	84.3
J80	0	72	NO	3.0	85.4	85.4	0.0	84.2	84.2
PHYSICAL							0.1		0.1

Samples: 120 System: J80 Date: 09/29/04 Time: 15.25.00 Range: 120 S

Partition: J80 2066 Model 004
 CPC Capacity: 108 Weight % of Max: 40.0 4h MSU Average: 59
 Image Capacity: 108 WLM Capping %: **** 4h MSU Maximum: 77

Partition	MSU Def	MSU Act	Cap Def	Proc Num	Logical Effect	Util % Total	- Physical LPAR	Util % Effect	- Physical Total
*CP							0.1	78.7	78.9
J80	0	85	NO	4.0	78.7	78.8	0.1	78.7	78.8
PHYSICAL							0.1		0.1

Figure 56. Example RMF workload activity report in WLM goal mode

Appendix C. Some of our Linux for zSeries samples, scripts and EXECs

This section lists some of the scripts and EXECs we did with in Chapter 18, "About our Linux virtual server environment," on page 261.

Files on our FTP server

Following are some sample files for our FTP server.

lticIPsetup

```
#!/bin/bash
#
# simple script to change the ifcfg-eth0 ip address and hostname regardless on Linux #
iplist=/etc/ip.list

if [ -e /etc/run_lticIPsetup ]; then

    # assuming first column in the iplist is the ip-address
    uniqueid=$(cat /proc/sysinfo | grep "VM00 Name:" | awk '{print $3}')

    oldhostname=ltic0000.pdl.pok.ibm.com
    oldip=192.168.70.170

    myhostname=${uniqueid}.pdl.pok.ibm.com
    myip=$(grep -i $uniqueid $iplist | awk '{print $1}')

    if [ $(grep -i -c "SUSE" /etc/issue) -gt 0 ]; then
        # we have suse system
        targetfile=$(grep -l $oldip /etc/sysconfig/network/*)
        hostfile=/etc/HOSTNAME
    elif [ $(grep -i -c "RED HAT" /etc/issue) -gt 0 ]; then
        # we have redhat system
        targetfile=$(grep -l $oldip /etc/sysconfig/network-scripts/*)
        hostfile=/etc/sysconfig/network
    else
        echo "system not recognized"
        exit
    fi

    # change network file permanently for future boot up
    cat $targetfile | sed "s/$oldip/$myip/" > /tmp/target-eth0
    cat $hostfile | sed "s/$oldhostname/$myhostname/" > /tmp/target-host
    mv /tmp/target-eth0 $targetfile
    mv /tmp/target-host $hostfile
    chmod 644 $targetfile
    chmod 644 $hostfile

    # change running ip now
    ifconfig eth0 $myip netmask 255.255.255.0 broadcast 192.168.70.255

    route add default gw 192.168.70.1
    hostname $myhostname

    # remove so that this script doesn't get run more than once
    rm /etc/run_lticIPsetup
fi
```

ip.list

```
192.168.70.170 LTIC0000
192.168.70.171 LTIC0001
192.168.70.172 LTIC0002
192.168.70.173 LTIC0003
192.168.70.174 LTIC0004
```

Linux for zSeries scripts and EXECs

```
192.168.70.175 LTIC0005
192.168.70.176 LTIC0006
192.168.70.177 LTIC0007
192.168.70.178 LTIC0008
192.168.70.179 LTIC0009
192.168.70.180 LTIC0010
```

Files on our USER 194 disk

Following are our samples for our USER 194 disk.

PROFILE EXEC

```
/* LINUX PROFILE EXEC */
USR = USERID()
CP SET PF12 RETRIEVE
CP SET MSG ON
CP SET EMSG ON
CP SET RUN ON
CP SET RETRIEVE MAX
CP TERM CHARDEL OFF
ACC 592 K

/* Check if there is a 200 swap disk, if not, create one from V-DISK */
'PIPE CMS Q V 200'
if RC <> 0 THEN 'SWAPGEN 200 2048000 diag'

/* Ipl the Linux system if Disconnected */
'PIPE CP Q ' USR ' | STACK LIFO '
PARSE PULL USER DASH STATE
IF STATE = 'DSC' THEN 'IPL 201 CLEAR'
ELSE call 'WELCOME'
EXIT
```

WELCOME EXEC

```
/******
/* WELCOME EXEC */
/* Display a Menu for the user */
/******
/* Ipl the Linux system if Disconnected */
'PIPE CMS ID | STACK LIFO '
PARSE PULL USER AT SYS .

TOP:
CLRSCRN /* CLEAR THE SCREEN */
say ' Welcome to ' USER ' on ' sys
say ' You have the following options: '
say;
say ' 1) IPL Linux on the 201 disk '
say ' 2) IPL Linux on the a disk other then 201 '
say ' 3) Install a new linux system '
say ' 4) Copy a pre-built Linux system '
say ' 5) Display my Networking Information'
say ' 6) What can I do with a LTICxxxx system '
say ' 7) EXIT '
say;
Pull opt
SELECT
    when opt = 1 then do
        'IPL 201 CLEAR'
    end

    when opt = 2 then do
        say 'What disk do you want to IPL'
        pull disk
```

```

        if strip(disk)='CMS' then 'IPL CMS'
        ELSE 'IPL ' disk ' CLEAR'
    end

    when opt = 3 then do
        call DISTRO
    end

    when opt = 4 then do
        call DISTCOPY
        SIGNAL TOP
    end

    when opt = 5 then do
        'IPDATA'
        say ; say 'Hit enter to continue'
        pull .
        SIGNAL TOP
    end

    when opt = 5 then do
        'IPDATA'
        say ; say 'Hit enter to continue'
        pull .
        SIGNAL TOP
    end

    when opt = 6 then do
        'HELP LTIC'
        SIGNAL TOP
    end
    otherwise nop;
end /* Select */

```

Files on our USER 195 disk

Following are our samples for our USER 195 disk.

DISKCOPY EXEC

```

/*****
/* DISKCOPY - Copy the DISTRO from the Master pack to 201 disk */
/*
/*
/*
/*
/*****
'CLRSCRN' /* Clear the screen */
'PIPE < DISKCOPY LIST * |spec 1-3 1 14-80 5 | CONSOLE'
say;say;
say 'Enter the ID of the system you would like to copy'
say '      or Blank to exit'
pull id
if id = '' then exit
'PIPE < DISKCOPY LIST * | LOCATE 1-3 /'id'/ | STACK LIFO '
PARSE PULL new_id VOL DESC

'CLRSCRN' /* Clear the screen */

"FLASHCOPY " VOL" 0 END 201 0 END"
IF RC = 0 then do
    say" The copy has completed"
    say ' Hit enter to return to the Main Menu'
    pull .
    EXIT 0
END

```

Linux for zSeries scripts and EXECs

```
/* If we got here the flash copy failed so we will do a DDR copy */
'CLRSCRN' /* Clear the screen */
SAY;say; SAY '                               HIT ENTER WHEN COPY COMPLETES '

    'makebuf'
    push ' '
    push 'YES'
    push 'YES'
    push 'COPY ALL'
    push 'OUTPUT 201 DASD'
    push 'INPUT 'VOL 'DASD'
    push 'SYSPRINT CONS'
    'DDR'
    'dropbuf'
    Say ' Copy has completed with Return Code: 'rc
    say ' Hit enter to return to the Main Menu'
    pull .
```

DISKCOPY LIST

ID	DISTRO
1 572A	SUSE SLES 9 RC5 64 BIT 2.6.5-7.97-s390x
2 5720	SUSE SLES 9 RC5 31 BIT 2.6.5-7.97-s390
3 572B	RH44 UPDATE 1 BETA 64 BIT 2.6.9-6.37.EL
4 5721	REDHAT RH44 BETA 1 31 BIT 2.6.8-1.528.2.5
5 572C	REDHAT RH44 BETA1 REFRESH 64 BIT 2.6.8-1.528.2.5
6 5722	REDHAT RH44 BETA1 REFRESH 31 BIT 2.6.8-1.528.2.5
7 572D	REDHAT RH44 BETA 2 64 BIT 2.6.9-1.648_EL
8 5723	REDHAT RH44 BETA 2 31 BIT 2.6.9-1.648_EL
9 5724	REDHAT RH43 UPDATE 3 31 BIT 2.4.21-4.EL

DISTRO EXEC

```
/******
/* DISTRO EXEC */
/*
/* This exec will display the linux RamDisk systems that can be
/* be loaded and then prompt the user for the system to install.*/
/*
/*
/******

'CLRSCRN' /* Clear the screen */
'PIPE < DISTRO LIST * |spec 1-3 1 14-80 5 | CONSOLE'
say;say;
say 'Enter the ID of the system you would like to install'
say ' or Blank to exit'
pull id
if id = '' then exit
'PIPE < DISTRO LIST * | LOCATE 1-3 /'id'/ | STACK LIFO '
PARSE PULL new_id TAG DESC
'LOADRDR ' tag
```

DISKCOPY LIST

ID	DISTRO
1 26579731	SUSE SLES 9 RC5 31 BIT 2.6.5-7.97-s390
2 26579764	SUSE SLES 9 RC5 64 BIT 2.6.5-7.97-s390x
3 24211764	REDHAT RH43 update 3 64 BIT- 2.4.21-17.EL
4 24211731	REDHAT RH43 update 3 31 BIT 2.4.21-17.EL

5	241931	SUSE	SLES 8	31 BIT	2.4.19-3suse-SMP
6	SLES864	SUSE	SLES 8	64 BIT	2.4.19
7	RHEL4U1B	REDHAT	RHEL4 UPDATE1 Beta	64 BIT	2.6.9-6.37.EL
8	RH4U131	REDHAT	RHEL4 UPDATE1 Beta	31 BIT	2.6.9-1.37.EL
9	26571264	SUSE	SLES 9 RC1-update	64 BIT	2.6.5-7.127-s390x
A	RH4RC231	REDHAT	RHEL4 RC2	31 BIT	2.6.9-1.906_EL
B	RH4RC264	REDHAT	RHEL4 RC2	64 BIT	2.6.9-1.906_EL
C	RH4RC_31	REDHAT	RHEL4 GA	31 BIT	2.6.?
D	RH4RC_64	REDHAT	RHEL4 GA	64 BIT	2.6.?
E	S8SP3_31	SuSE	SLES 8 SP 3	31 BIT	2.6.?
F	S8SP3_64	SuSE	SLES 8 SP 3	64 BIT	2.6.?

IPDATA EXEC

```

/*****/
/* IPDATA EXEC - Display the users IP Information.          */
/*                                                         */
/*****/
'PIPE CMS ID | STACK LIFO '
PARSE PULL USER AT SYS .
CLRSCRN /* CLEAR THE SCREEN */

'PIPE < IPDATA LIST * | LOCATE 1-8 /'USER'/'| VAR OUTPUT '
PARSE VAR OUTPUT USERID IP

say ' The IP Address for user: ' USER ' on ' sys 'is: 'IP

/* GET A LIST OF ALL THE VSWITCHES */
'PIPE CP Q VSWITCH | LOCATE /VSWITCH SYSTEM/ | STEM VSWITCH. '

x=1

DO VSWITCH.0
  switch_str=vswitch.x
  parse var switch_str . . switch .
  'PIPE CP Q VSWITCH DETAIL 'switch'| LOCATE /RDEV/ | var port'
  PARSE VAR PORT . name .
  'PIPE CP Q VSWITCH DETAIL 'switch' | LOCATE /'USER'/ | var detail '
  PARSE VAR detail . . . . dev .
  if dev <> '' then do
    say; say ' VSWITCH 'switch' DEVICE:'dev ' Portname:'name
  end
  x=x+1
END /* do */

```

IPDATA LIST

```

LTIC0000    192.167.70.170
LTIC0001    192.167.70.171
LTIC0002    192.167.70.172
LTIC0003    192.167.70.173
LTIC0004    192.167.70.174
LTIC0005    192.167.70.175
LTIC0006    192.167.70.176
LTIC0007    192.167.70.177
LTIC0008    192.167.70.178
LTIC0009    192.167.70.179
LTIC0010    192.167.70.180

```

LOADRDR EXEC

```
/******  
/* LOADRDR EXEC */  
/* This EXEC will load the required files into the reader */  
/* so that you can IPL the LINUX RAM DISK to build the DISTRO */  
/* */  
/* Usage: DISKLOAD linux_distro_id */  
/* */  
/******  
arg tag  
if tag = '' then do  
    say 'USEAGE:  linux_distro_id '  
    say;  
say 'RUN the xxxx EXEC to get the linux_distro_id'  
EXIT  
end  
  
'close rdr'  
'purge rdr all'  
'spool punch * rdr'  
  
/* Need to add some code to make sure the files exist */  
  
'PUNCH 'tag ' IMAGE D (NOH'  
'PUNCH 'tag ' PARM D (NOH'  
'PUNCH 'tag ' INITRD D (NOH'  
'change rdr all keep nohold'  
'ipl 00c clear'
```

Appendix D. Availability of our test reports

The following information describes the variety of ways in which you can obtain our test reports.

Our publication schedule is changing

Starting in 2003, our publication schedule changed somewhat from our traditional quarterly cycle as a result of the planned change in the development cycle for annual z/OS releases. Keep an eye on our Web site for announcements about the availability of new editions of our test report.

Availability on the Internet: You can view, download, and print the most current edition of our test report from our z/OS Integration Test Web site at:

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

Our Web site also provides all of our previous year-end editions, each of which contains all of the information from that year's interim editions.

You can also find our test reports on the z/OS Internet Library Web site 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 BookServer. 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 free of charge from the IBM Softcopy Web site at www.ibm.com/servers/eserver/zseries/softcopy/.

- Adobe Portable Document Format (PDF)

PDF documents require the Adobe Acrobat Reader to view and print. Your Web browser can invoke the Acrobat Reader to work with PDF files online. You can also download PDF files and access them locally using the Acrobat Reader. You can get the Acrobat Reader free of charge from www.adobe.com/products/acrobat/readstep.html.

Softcopy availability: BookMaster BOOK and Adobe PDF versions of our test reports are included in the OS/390 and z/OS softcopy collections on CD-ROM and DVD. For more information about softcopy deliverables and tools, visit the IBM Softcopy Web site (see above for the Web site address).

Availability of our test reports

A note about the currency of our softcopy editions

Because we produce our test reports toward the end of the product development cycle, just before each new software release becomes generally available (GA), we cannot meet the production deadline for the softcopy collections that coincide with the product's GA release. 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.

Hardcopy availability: Our December 2001 edition was the last edition to be published in hardcopy. As of 2002, we no longer produce a hardcopy edition of our year-end test reports. You can still order a printed copy of a previous year-end edition (using the order numbers shown in Table 18 below) through your normal ordering process for IBM publications.

Available year-end editions: The following year-end editions of our test report are available:

Table 18. Available year-end editions of our test report

Title	Order number	Covers our test experiences for...		Softcopy collection kits
		This year...	And these releases...	
<i>zSeries Platform Test Report</i>	SA22-7997-00	2004	z/OS V1R6	SK3T-4269-14
<i>z/OS Parallel Sysplex Test Report</i>	SA22-7663-09	2003	z/OS V1R4	SK3T-4269-11 SK3T-4271-11
<i>z/OS Parallel Sysplex Test Report</i>	SA22-7663-07	2002	z/OS V1R3 and V1R4	SK3T-4269-07 SK3T-4271-07
<i>z/OS Parallel Sysplex Test Report</i>	SA22-7663-03	2001	z/OS V1R1 and V1R2	SK3T-4269-03 SK3T-4270-04 SK3T-4271-03
<i>OS/390 Parallel Sysplex Test Report</i>	GC28-1963-19	2000	OS/390 V2R9 and V2R10	SK2T-6700-24 SK2T-6718-14
<i>OS/390 Parallel Sysplex Test Report</i>	GC28-1963-15	1999	OS/390 V2R7 and V2R8	SK2T-6700-17
<i>OS/390 Parallel Sysplex Test Report</i>	GC28-1963-11	1998	OS/390 V2R5 and V2R6	SK2T-6700-15 and -17
<i>OS/390 Parallel Sysplex Test Report</i>	GC28-1963-07	1997	OS/390 V1R3 and V2R4	SK2T-6700-11 and -13
<i>OS/390 Parallel Sysplex Test Report</i>	GC28-1963-03	1996	OS/390 V1R1 and V1R2	SK2T-6700-07
<i>S/390 MVS Parallel Sysplex Test Report</i>	GC28-1236-02	1995	MVS/ESA SP V5	none

Other related publications: From our Web site, you can also access other related publications, including our companion publication, *OS/390 Parallel Sysplex Recovery*, GA22-7286, as well as previous editions of *OS/390 e-Business Integration Test (eBIT) Report*.

Appendix E. Useful Web sites

We have cited the IBM books we used to do our testing as we refer to them in each topic in this test report. This chapter contains listings of some of the Web sites that we reference in this edition or previous editions of our test report.

IBM Web sites

Table 19 lists some of the IBM Web sites that we reference in this edition or previous editions of our test report:

Table 19. Some IBM Web sites that we reference

Web site name or topic	Web site address
<i>IBM Terminology</i> (includes the <i>Glossary of Computing Terms</i>)	www.ibm.com/ibm/terminology/
<i>IBM CICS Transaction Gateway</i>	www.ibm.com/software/ts/cics/library/
<i>IBM HTTP Server library</i>	www.ibm.com/software/websphere/httpservers/library.html
<i>IBMLink™</i>	www.ibm.com/ibmlink/
<i>IBM mainframe servers Internet library</i>	www.ibm.com/servers/eserver/zseries/library/
<i>IBM Redbooks</i>	www.ibm.com/redbooks/
<i>IBM Systems Center Publications</i> (IBM TechDocs — flashes, white papers, etc.)	www.ibm.com/support/techdocs/
<i>Linux at IBM</i>	www.ibm.com/linux/
<i>Net.Data Library</i>	www.ibm.com/software/data/net.data/library.html
<i>OS/390 Internet library</i>	www.ibm.com/servers/s390/os390/bkserv/
<i>Parallel Sysplex</i>	www.ibm.com/servers/eserver/zseries/psa/
<i>Parallel Sysplex Customization Wizard</i>	www.ibm.com/servers/eserver/zseries/zos/wizards/parallel/
<i>System Automation for OS/390</i>	www.ibm.com/servers/eserver/zseries/software/sa/
<i>WebSphere Application Server</i>	www.ibm.com/software/webervers/appserv/
<i>WebSphere Application Server library</i>	www.ibm.com/software/webervers/appserv/zos_os390/library/
<i>WebSphere Studio library</i>	www.ibm.com/software/websphere/studio/library.html
<i>WebSphere Studio Workload Simulator</i>	www.ibm.com/software/awdtools/studioworkloadsimulator/library/
<i>z/OS Consolidated Service Test</i>	www.ibm.com/servers/eserver/zseries/zos/servicetst
<i>z/OS downloads</i>	www.ibm.com/servers/eserver/zseries/zos/downloads/
<i>z/OS Integration Test</i> (includes information from OS/390 Integration Test and e-business Integration Test (ebIT))	www.ibm.com/servers/eserver/zseries/zos/integtst/
<i>z/OS Internet library</i>	www.ibm.com/servers/eserver/zseries/zos/bkserv/
<i>z/OS UNIX System Services</i>	www.ibm.com/servers/eserver/zseries/zos/unix/
<i>z/OS.e home page</i>	www.ibm.com/servers/eserver/zseries/zose/

Table 19. Some IBM Web sites that we reference (continued)

Web site name or topic	Web site address
<i>z/OS.e Internet library</i>	www.ibm.com/servers/eserver/zseries/zose/bkserv/

Other Web sites

Table 20 lists some other non-IBM Web sites that we reference in this edition or previous editions of our test report:

Table 20. Other Web sites that we reference

Web site name or topic	Web site address
<i>Cisco Systems</i>	www.cisco.com/
<i>Java Servlet Technology</i>	java.sun.com/products/servlet/
<i>Java 2 Platform, Enterprise Edition (J2EE)</i>	java.sun.com/products/j2ee/
<i>JavaServer Pages (JSP)</i>	java.sun.com/products/jsp/
<i>J2EE Connector Architecture</i>	java.sun.com/j2ee/connector/
<i>SUSE Linux</i>	www.suse.com/

Appendix F. 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* 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/

One exception is command syntax that is published in railroad track format; screen-readable copies of z/OS books with that syntax information are separately available in HTML zipped file form upon request to mhvrdfs@us.ibm.com.

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.

Notices

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	NetView
BatchPipes	Notes
BookManager	OS/2
BookMaster	OS/390
CICS	Parallel Sysplex
CICSplex	PR/SM
DB2	Processor Resource/Systems Manager
DB2 Connect	QMF
DFS	RACF
DFSMS/MVS	RAMAC
DFSMSHsm	Redbooks
DFSMSrmm	RMF
Enterprise Storage Server	RS/6000
ESCON	S/390
@server	SP
FICON	SupportPac
IBM	Sysplex Timer
ibm.com	Tivoli
IBMLink	TotalStorage
IMS	VisualAge
Infoprint	VSE/ESA
Language Environment	VTAM
Library Reader	WebSphere
MQSeries	z/OS
MVS	z/OS.e
MVS/ESA	z/VM
Net.Data	zSeries

The following terms are trademarks of other companies:

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

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

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

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

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

Index

A

- accessibility 323
- application enablement
 - configuration 123
 - workloads 132
- ARM enablement 134
- ATM LAN emulation
 - configuration 125
- automation
 - See also* Parallel Sysplex automation
 - Parallel Sysplex 109
- availability
 - of this document 319

C

- CFCC
 - dispatcher rewrite testing 11
- channel connectivity
 - coupling facility channels 11
- channel subsystem
 - coupling facility channels 13
 - CTC channels 13
 - ESCON channels 13
 - FICON channels 13
- CICS TS 2.3
 - migrating to 59
 - migration experiences 63
 - overview of migration 59
 - performing the migration 60
 - migrating CICSplex SM 61
 - migrating the CASs 61
 - migrating the CMASs 62
 - migrating the MASs 63
 - preparing for migration 60
- Cloning Linux images on z/VM 5 263
- cloning system
 - preparing the VM environment 264
- common DASD
 - defining 264
- configuration
 - application enablement 123
 - ATM LAN emulation 125
 - coupling facility channels 11
 - Ethernet LAN 124
 - hardware details 7
 - mainframe servers 7
 - hardware overview 5
 - ipV6 Environment 125
 - LDAP Server overview 173
 - networking 123
 - Parallel Sysplex hardware 5
 - sysplex hardware details
 - coupling facilities 9
 - other sysplex hardware 12
 - sysplex software 16
 - token-ring LAN 127

- configuration (*continued*)
 - VTAM 19
 - WebSphere Application Server for z/OS 231
- console restructure
 - testing 117

D

- DB2
 - migrating to V8 with LDAP 204
- DB2 UDB JCC Connectors
 - using 235
- DB2 V7.2 FP8 Server
 - installing 291
- DB2 V8
 - enabling new function mode 86
 - migrating 65
 - migrating first member to compatibility mode 69
 - migrating remaining members to compatibility mode 77
 - migrating to new function mode 82
 - migration considerations 65
 - premigration activities 66
 - preparing for new function mode 82
 - running in new function mode 88
 - V7 coexistence issues 77
 - verifying the installation 89
- DB2 V8.1
 - configuring clients on Linux on zSeries 278
- directory profile
 - LTICPRO 265
- disability 323
- DISKCOPY EXEC 265
- DISKCOPY LIST 265
- distribution
 - of this document 319
- DISTRO EXEC 265
- DISTRO LIST 265
- Domino Administrator Client
 - installing on Windows 284
- Domino Mail Server V6.5.4
 - installing and running on Linux on zSeries 282
- Domino server
 - installing on Linux on zSeries 283
- Domino Server
 - starting on the Linux on zSeries 284
- DVIPA 134
- dynamic enablement
 - relation to IBM License Manager 16
 - relation to IFAPRDxx parmlib member 16

E

- Enterprise Identity Mapping 251
 - using client authentication with digital certificates 251
 - using CRAM-MD5 password protection 254

- Enterprise Identity Mapping (*continued*)
 - using Kerberos authentication 252
- environment
 - networking and application enablement 123
 - Parallel Sysplex 5
 - WebSphere Application Server for z/OS 231
 - workloads 20
- ESCON channels 13
- Ethernet
 - 10BASE-T 124
 - Fast Ethernet 124
 - Gigabit Ethernet 124
- Ethernet LAN
 - configuration 124
- EXECs
 - Linux for zSeries 313

F

- FICON channels 13
 - FICON native (FC) mode 13
- FLASHCOPY command
 - adding to the "Z" class 264
- FTP
 - server enablement with Kerberos 214, 217
 - testing with Kerberos 217
 - with Kerberos 214

H

- hardware
 - configuration details 7
 - mainframe servers 7
 - configuration overview 5
 - coupling facility channel configuration 11
 - Parallel Sysplex configuration 5
- hardware crypto acceleration
 - using on the Apache2 server 275
- HTTP server
 - See also* IBM HTTP Server
 - configuring 274
- HTTPS
 - enabling on the Apache2 server 274
 - enabling on the IBM HTTP server 274
 - enabling using hardware crypto acceleration on the Apache2 server 275

I

- IBM HTTP Server
 - changes to certificate management 171
 - WebSphere troubleshooter 171
- IBM Tivoli Directory Server 5.2 server/client
 - See also* LDAP Server
 - setting up SSL client and server authentication
 - between z/OS LDAP V1R6 server/client 192
- IFAPRDxx parmlib member
 - relation to dynamic enablement 16
- IMS
 - CSL performance considerations 101
 - IMS Connect 95

- IMS (*continued*)
 - migrating to IMS V9 93
- IMS Connect
 - IRLM 2.2 migration 96
 - migrating to IMS V9 95
- Integrated Cryptographic Service Facility (ICSF)
 - using 18
- IPDATA EXEC 265
- IPDATA LIST 265
- iptables
 - configuring 286
- ipV6 Environment Configuration
 - configuration 125
- IRLM 2.2
 - migrating from IRLM 2.1 96
- ISV security products
 - TrendMicro's ScanMail 297
 - TrendMicro's ServerProtect 298

K

- Kerberos 211
 - configuring a Linux workstation 216
 - FTP 214
 - FTP server enablement 214, 217
 - testing FTP 217
 - working with principals in RACF 217
- key files
 - defining on common DASD 264
- keyboard 323

L

- LANs
 - LAN A description 128
 - LAN B description 129
 - LAN C description 130
 - token-ring backbone description 128
- LDAP
 - setting up SSL client and server authentication
 - between IBM Tivoli Directory Server 5.2 server/client 192
 - between Sun ONE Directory Server 5.2 server/client 186
- LDAP Server 173
 - configuration overview 173
 - migrating to DB2 V8 204
- Linux
 - security 269
- Linux for zSeries
 - EXECs 313
 - scripts 313
- Linux images on z/VM 5
 - cloning 263
- Linux on zSeries
 - configuring DB2 V8.1 clients 278
 - future projects 303
 - IBM security products 290
 - installing and running Domino Mail Server V6.5.4 282
 - installing the Domino server 283

- Linux on zSeries *(continued)*
 - installing WebSphere Application Server 273
 - installing WebSphere Application Server Network Deployment V5.1 273
 - ISV security products 297
 - middleware environment 273
 - network configuration 270
 - open source security products 286
 - planning our environment 269
 - security testing 301
 - starting the Domino Server 284
 - where to find more information 303
- Linux virtual servers 261
- LookAt message retrieval tool xxi
- LTICPRO directory profile
 - including 265
- LTICxxx
 - defining the directory entry 266
- LTICxxxx
 - Setting the IP and HOSTNAME 266
 - setting up 266
 - verifying the setup 267

M

- message retrieval tool, LookAt xxi
- migrating
 - DB2 V8 65
- MQSeries
 - See WebSphere Business Integration
- msys for Operations
 - See Parallel Sysplex automation

N

- naming conventions
 - CICS and IMS subsystem jobnames 18
- Network Authentication Service for z/OS 211
- network configuration
 - Linux 270
- networking
 - configuration 123
 - ATM LAN emulation 125
 - Ethernet LAN 124
 - ipV6 Environment 125
 - token-ring LAN 127
 - workloads 132
- NFS
 - migrating to the OS/390 NFS 132
 - preparing for system outages 133
 - recovery 133
- NFS environment
 - acquiring DVIPA 134
 - setting up ARM 134
- Notices 325

O

- On/Off Capacity on Demand Testing
 - RMF workload activity reports 310

- open source security products
 - Linux on zSeries 286
- OSA-2
 - ATM feature 123
 - ENTR feature 123, 124, 127
 - FENET feature 123, 124
- OSA-Express
 - ATM feature 123
 - FENET feature 123, 124
 - Gigabit Ethernet 123
 - Gigabit Ethernet feature 124

P

- Parallel Sysplex
 - hardware configuration 5
- Parallel Sysplex automation 109
- parmlib members
 - related to z/OS V1R4 and z/OS.e V1R4 305
- performance
 - See also RMF
 - considerations for IMS CSL 101
 - RMF reports 307
 - z/OS 52
- principals
 - working with Kerberos in RACF 217

R

- RACF
 - working with Kerberos principals 217
- Recovery
 - preparing for with NFS 133
- Red Hat Enterprise Linux 3 Update 4
 - defining Samba 281
- RMF
 - Monitor I Post Processor Summary Report 307
 - Monitor III Online Sysplex Summary Report 307
 - Workload Activity Report in WLM Goal Mode 308
 - workload activity report with our On/Off Capacity on Demand Testing 310

S

- SA OS/390
 - See Parallel Sysplex automation
- Samba
 - defining on Red Hat Enterprise Linux 3 Update 4 281
- scripts
 - Linux for zSeries 313
- security
 - Linux 269
- security products
 - IBM security products for Linux on zSeries 290
 - ISV security products for Linux on zSeries 297
- Security Server LDAP Server
 - See LDAP Server
- Security Server Network Authentication Service for z/OS 211
 - peer trust between z/OS and Windows 2000 211

- security testing
 - on Linux on zSeries 301
- shortcut keys 323
- SLES 8 Linux clients
 - authenticating 296
- software
 - configuration overview 16
 - sysplex configuration 16
- SSL tunneling
 - setting 279
- su command
 - changing TSO identity 169
- Sun ONE Directory Server 5.2 server/client
 - setting up SSL client and server authentication between z/OS LDAP V1R6 server/client 186

T

- TAM WebSeal
 - authenticating and authorizing Web transactions 295
- tasks
 - migrating to z/OS
 - overview 29
 - migrating to z/OS V1R5 35
 - high-level migration process 35
 - other migration activities 36
 - migrating to z/OS V1R6 29
 - high-level migration process 29
 - other migration activities 30
 - migrating to z/OS.e V1R5 48
 - high-level migration process 48
 - other migration activities 49
 - migrating to z/OS.e V1R6 31
 - high-level migration process 31
 - other migration activities 33
- Tivoli Access Manager
 - authenticating and authorizing Web transactions 295
- Tivoli Risk Manager
 - planning and installing 290
- token-ring LAN
 - backbone description 128
 - configuration 127
 - LAN A description 128
 - LAN B description 129
 - LAN C description 130
- TrendMicro's ScanMail
 - installing 297
- TrendMicro's ServerProtect
 - installing 298
- TRM Event Server
 - installing 291

U

- UNIX
 - See z/OS UNIX System Services
- URLs
 - referenced by our team 321
- USER 194 Disk 264

- USER 195 Disk 265
 - userid "USER"
 - defining 264

V

- VTAM
 - configuration 19

W

- WBIMB 23
- Web sites
 - used by our team 321
- WebSphere Application Server Edge Component Caching Proxy V5.1
 - setting SSL tunneling on 279
- WebSphere Application Server for z/OS
 - CICS Transaction Gateway Connector V5.1 in local mode 236
 - Enabling Global Security and SSL 236
 - Migrating JDBC from DB2 V7 to DB2 V8 235
 - Migrating to WebSphere for z/OS V5.0
 - our naming conventions 234
 - where to find more information 248
 - Migrating to WebSphere for z/OS V5.X
 - test and production configurations 232
 - Web application workloads 233
 - our test environment 231
 - current software products and release levels 231
 - using 231
 - Using DB2 UDB JCC Connectors 235
- WebSphere Application Server ND Edge Component Load Balancer V5.1
 - configuring 280
- WebSphere Business Integration 219
 - shared channels in a distributed-queuing management environment 222
 - shared channel configuration 223
 - testing shared channel recovery 224
 - shared queues and coupling facility structures
 - coupling facility structure configuration 219
 - using shared queues and coupling facility structures 219
 - queue sharing group configuration 219
 - recovery behavior of queue managers and coupling facility structures 220
- WebSphere Business Integration Message Broker 226
 - _BPXK_MDUMP environment variable 226
 - applying Fix Pack 02 and 03 229
 - migrating to Version 5.0 228
 - resolving a EC6-FF01 abend 228
 - testing WMQI V2.1 on DB2 V8 226
 - useful WBIMB Web sites 229
 - writing dumps to MVS data sets 226
- WebSphere Business Integration Message Broker 23
- WebSphere MQ
 - See WebSphere Business Integration
- WebSphere MQ workloads 22
- WebSphere troubleshooter 171

- workload
 - application enablement 21
 - automatic tape switching 21
 - base system functions 20
 - database product 25
 - DB2 batch 26
 - DB2 data sharing 26
 - IMS data sharing 25
 - networking 24
 - networking and application enablement 132
 - sysplex batch 26
 - sysplex OLTP 25
 - VSAM/NRLS 26
 - VSAM/RLS data sharing 26
- workloads 20
 - WebSphere MQ 22

Z

- z/OS
 - performance 52
 - summary of new and changed parmlib members for z/OS V1R4 and z/OS.e V1R4 305
- z/OS Distributed File Service
 - zFS enhancements in z/OS V1R6 165
 - zFS performance monitoring 166
- z/OS Security Server LDAP Server
 - See LDAP Server
- z/OS UNIX System Services 137
 - enhancements in z/OS V1R5 137
 - enhancements in z/OS V1R6 145
 - HFS to zFS automount migration 164
 - managing a hierarchical file system (HFS) 164
 - managing a zSeries file system (zFS) 165, 169
 - su command
 - changing TSO identity 169
- z/OS V1R5
 - high-level migration process 35
 - applying coexistence service 35
 - IPLing additional z/OS V1R5 images 36
 - IPLing the first z/OS V1R5 image 35
 - updating parmlib 35
 - updating RACF templates 36
 - other migration activities 36
 - recompiling REXX EXECs for automation 37
 - running with mixed product levels 36
 - using concatenated parmlib 36
- z/OS V1R6
 - high-level migration process 29
 - applying coexistence service 30
 - IPLing additional z/OS V1R6 images 30
 - IPLing the first z/OS V1R6 image 30
 - updating parmlib 30
 - updating RACF templates 30
 - other migration activities 30
 - recompiling REXX EXECs for automation 31
 - running with mixed product levels 31
 - using concatenated parmlib 31
- z/OS.e V1R5
 - high-level migration process 48
 - IPLing the system 49

- z/OS.e V1R5 (*continued*)
 - high-level migration process (*continued*)
 - obtaining licenses for z/OS.e 48
 - updating our IEASYMPT member 49
 - updating our LOADxx member 49
 - updating parmlib 49
 - updating the LPAR name 49
 - other migration activities 49
 - LPAR environment 49
 - removing from MNPS 51
 - removing from TSO generic resource groups 51
 - updating our ARM policy 50
 - using concatenated parmlib 49
 - using current levels of JES2 and LE 50
 - other migration experiences 51
- z/OS.e V1R6
 - high-level migration process 31
 - IPLing the system 33
 - obtaining licenses for z/OS.e 32
 - updating our IEASYMPT member 33
 - updating our LOADxx member 32
 - updating parmlib 32
 - updating the LPAR name 32
 - other migration activities 33
 - LPAR environment 33
 - removing from MNPS 34
 - removing from TSO generic resource groups 34
 - updating our ARM policy 34
 - using concatenated parmlib 33
 - using current levels of JES2 and LE 34
 - other migration experiences 35
- z/VM 5
 - Cloning Linux images on 263
- zSeries Hardware Cryptographic Acceleration
 - Web Servers 274

Readers' Comments — We'd Like to Hear from You

z/OS

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

Version 1 Release 6

Publication No. SA22-7997-01

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	<input type="checkbox"/>				

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	<input type="checkbox"/>				
Complete	<input type="checkbox"/>				
Easy to find	<input type="checkbox"/>				
Easy to understand	<input type="checkbox"/>				
Well organized	<input type="checkbox"/>				
Applicable to your tasks	<input type="checkbox"/>				

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? Yes No

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.

Name

Address

Company or Organization

Phone No.



Fold and Tape

Please do not staple

Fold and Tape



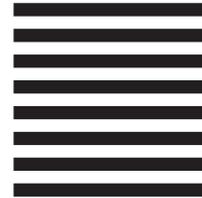
NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

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



Printed in USA

SA22-7997-01

