

z/OS



Parallel Sysplex Test Report

V1R3

z/OS



Parallel Sysplex Test Report

V1R3

Note!

Before using this information and the products it supports, be sure to read the general information under "Notices" on page 219.

Sixth Edition, June 2002

This is a major revision of SA22-7663-04.

This edition applies to Parallel Sysplex environment function that includes data sharing and parallelism. Parallel Sysplex uses the OS/390, z/OS, or z/OS.e 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, 2002. 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

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 xx.

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.

Michael Ackerbauer	John Corry	Sue Marcotte
Al Alexa	Don Costello	Tammy McAllister
Loraine Arnold	Kevin Coyne	James Mitchell
Tyrone Baird	Luis Cruz	Bob Muenkel
Ryan Bartoe	Tony DiLorenzo	Deron Nix
Frank Bellacicco	Bob Fantom	Brian O'Leary
Duane Beyer	Nancy Finn	David Quenzler
Jeff Bixler	Bobby Gardinor	Jim Rossi
Dave Bliss	Joan Kelley	Tom Sirc
Michael Brown	Fred Lates	Paul Sonnenberg
Dave Buehl	Al Lease	Jeff Stokes
Jon Burke	Frank Lefevre	Jim Stutzman
Alex Caraballo	Arvind Mistry	Sri Talluri
Phil Chan	Kristine Logan	

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 *z/OS Integration Test* Web site at:

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

Contents

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

Part 1. Parallel Sysplex 1

Chapter 1. About our Parallel Sysplex environment.	3
Overview of our Parallel Sysplex environment	3
Our Parallel Sysplex hardware configuration	3
Overview of our hardware configuration	3
Hardware configuration details.	5
Our Parallel Sysplex software configuration	12
Overview of our software configuration	12
About our naming conventions	14
Our VTAM configuration	14
Our workloads	15
Base system workloads.	16
Application enablement workloads.	17
Networking workloads	18
Database product workloads	19
Chapter 2. Migrating to and using z/OS	21
Overview	21
Migrating to z/OS V1R3	21
z/OS V1R3 base migration experiences.	21
Introducing z/OS.e V1R3	23
What is z/OS.e V1R3?	23
z/OS.e V1R3 base migration experiences	24
Other experiences with z/OS.e V1R3.	28
Summary of requirements for z/OS.e V1R3	28
z/OS performance.	29
Chapter 3. Using MQSeries for OS/390 V5R2	31
Implementing shared queues.	31
Task 9: Setting up the DB2 environment for shared queues	33
Task 10: Setting up the coupling facility structures for shared queues	37

Task 12: Customizing the initialization input data sets for shared queues	38
Task 15: Adding the MQSeries entries to the DB2 data-sharing group	39
Task 16: Tailor your system parameter module	40
Migrating our MQSeries-CICS bridge queues to shared queues	40
Migrating to a clustered MQSeries setup for SDSF	41
Steps for migrating to a clustered MQSeries setup for SDSF	41
Chapter 4. Implementing the DFSMSHsm common recall queue	43
Steps for defining the DFSMSHsm common recall queue environment	43
Using the DFSMSHsm common recall queue	43
Starting CRQ processing	44
Displaying and controlling CRQ processing	44
Experiences with CRQ processing	44
Chapter 5. Migrating to CICS TS Version 2 Release 2	45
Overview of migrating to CICS TS 2.2	45
Performing the migration to CICS TS 2.2	46
Preparing for migration	46
Migrating CICSplex SM	47
Migrating the CASs	47
Migrating the CMASs	48
Migrating the MASs	49
Experiences with migrating to CICS TS 2.2	49
Chapter 6. Parallel Sysplex automation	51

Part 2. Networking and application enablement 53

Chapter 7. About our networking and application enablement environment	57
Our networking and application enablement configuration	57
Our Ethernet LAN configuration	58
Our ATM configuration	59
Our token ring LAN configuration	59
Comparing the network file systems	64
Networking and application enablement workloads	65
Chapter 8. Using z/OS UNIX System Services	67
z/OS UNIX enhancements in z/OS V1R3	67
Using the automount enhancement	67
Using the enhanced confighfs command	67
Using the BPXPRMxx UNMOUNT option for file system mounts	68
Using OMVS restart	68
Starting a z/OS UNIX colony address space outside of JES	69
Monitoring the z/OS UNIX mount table limit	71
Using access control lists (ACLs)	74
Overview of access control lists	74
About ACL terminology	75
Some helpful tips	75
Working with access ACLs	76
Working with default ACLs	79
Auditing changes to ACLs	83
Using the getfacl and ls commands	86
Using the setfacl command	90
Using the find command with ACLs	94
Using the pax and tar commands with ACLs	99
Using copytree and the cp and mv commands with ACLs	102

Using the df and getconf commands to determine file system ACL support	104
Using UNIXPRIV class profiles with ACLs	106
Using ISHELL with ACLs	112
Managing a hierarchical file system (HFS)	115
Managing a zSeries file system (zFS)	115
zFS terminology	115
Setting up zFS	115
Chapter 9. Using IBM HTTP Server	121
Protecting IBM HTTP Server resources with LDAP	121
Customizing the HTTP server configuration file	121
Customizing the LDAP server configuration file	122
Verifying the LDAP protection for HTTP server resources	122
Chapter 10. Using LDAP Server	125
Overview of our LDAP configuration.	125
Enabling native authentication	126
Updating the schema for native authentication	126
Learning from our mistakes	127
Updating the slapd.conf configuration file	127
Modifying directory entries to perform native authentication	128
Enabling the IBM HTTP Server to exploit LDAP native authentication	130
Setting up a referral between LDAP databases	132
Setting up an LDAP server on Windows NT	132
Defining a referral in LDAP Server on z/OS	133
Testing the referral function	133
Enabling LDAP for Kerberos authentication	134
Experiences with enabling LDAP for Kerberos authentication	134
Resolving a problem with malformed distinguished names in directory entries	137
Resolving the problem.	137
Chapter 11. Setting up Security Server Network Authentication Service	139
Experiences setting up Network Authentication Service.	139
Missing HFS directories	139
Failure to consult the program directory for initial customization	140
Conflict with DCE	141
Chapter 12. Using Enhanced ASCII functionality	143
Overview of Enhanced ASCII	143
Understanding file tagging and automatic conversion	143
Implementing Enhanced ASCII	145
Working with Enhanced ASCII	146
Displaying file tag information	146
Controlling Enhanced ASCII functionality	147
Chapter 13. Using PKI Services	157
Introduction to PKI Services.	157
Installing and configuring prerequisite software.	157
Installing and configuring IBM HTTP Server	157
Installing and configuring OCSF and OCEP	158
Installing and configuring LDAP Server	159
Configuring the system for PKI Services	160
Running IKYSETUP to perform RACF administration	160
Configuring the UNIX runtime environment	162
Tailoring the LDAP configuration for PKI Services.	163
Updating the IBM HTTP Server configuration and starting the server	164

Tailoring the PKI Services configuration file for LDAP	166
Creating VSAM data sets and starting PKI Services	166
Customizing PKI Services	169
Customizing the end-user Web pages	169
Using PKI Services	170
Using the end-user Web pages	170
Chapter 14. Using IBM WebSphere Application Server for z/OS and OS/390	177
Packaging and deploying Web applications using WAR files	177
Steps for creating and deploying WAR files	177
Experiences with WAR files	178
Migrating Web applications to WebSphere Application Server V4.0 for z/OS and OS/390	180
Migrating Web applications to the WebSphere for z/OS V4.0 plugin	180
Enabling WebSphere Application Server V4.0 for z/OS and OS/390 on a sysplex	181
Planning for WebSphere for z/OS and sysplex	182
Preparing the security system	182
Setting up data sharing	182
Customizing base z/OS functions on other systems in the sysplex	182
Updating the TCP/IP configuration	183
Setting up LDAP files for other systems in the sysplex	183
Defining the second WebSphere for z/OS system	184
Running the installation verification programs (IVP)	186
Updating the configuration for servlets that access MOFW or EJB applications from WebSphere for z/OS V4.0 plugin	187
Updating the startup procedure for the HTTP server	187
Updating the httpd.conf file	188
Updating the httpd.envvars file	188
Updating the was.conf file	188
Accessing our servlet	188
Migrating to WebSphere Application Server V4.0.1 for z/OS and OS/390	188
Steps to migrate from WebSphere for z/OS V4.0 to V4.0.1	189
Additional migration experiences with WebSphere for z/OS V4.0.1	192
Where to find more information	192
Chapter 15. Running Linux on IBM mainframe servers	193
Setting up Linux guests under z/VM	193
Preparing the DASD	194
Creating a VM user definition for the first Linux guest	194
Setting up IP network connectivity	195
Building the first Linux instance	199
Creating and tailoring additional Linux instances	200
Setting up HiperSockets communications for LVMs	200
Defining the iQDIO hardware configuration	201
Adding HiperSockets links to the Linux virtual machine configuration	202
Configuring the network interface under Linux	203
Experiences with HiperSockets communications	205
Setting up LPAR CPU management for non-z/OS partitions	205
Steps for setting up LPAR CPU management for our z/VM partition	205
Appendix A. Some of our parmlib members	207
Appendix B. Some of our RMF reports	209
RMF Monitor I post processor summary report	209

RMF Monitor III online sysplex summary report	209
RMF workload activity report in WLM goal mode	210
Appendix C. Document availability and distribution.	213
Appendix D. Useful Web sites	215
IBM Web sites	215
Other Web sites	216
Appendix E. Accessibility	217
Using assistive technologies	217
Keyboard navigation of the user interface.	217
Notices	219
Trademarks.	221
Index	223

Figures

1.	Our sysplex hardware configuration	4
2.	Our coupling facility channel configuration	10
3.	Our sysplex software configuration	13
4.	Our VTAM configuration	15
5.	Our shared queue configuration	32
6.	Our CICS TS 2.2 and CPSM 2.2 configuration	45
7.	Our networking and application enablement configuration	57
8.	Our token-ring LAN A	61
9.	Our token-ring LAN B	62
10.	Our token-ring LAN C	63
11.	Mixed sysplex viewing HFS filesystem contents	118
12.	Mixed sysplex viewing HFS and zFS filesystem contents.	119
13.	Overview of our LDAP configuration	125
14.	Overview of HTTP server protection and LDAP native authentication	130
15.	Conceptual summary of file tagging and automatic conversion	144
16.	Automatic conversion for a C program performing I/O on an ASCII file.	145
17.	Summary of factors that control automatic conversion.	147
18.	Sample of one of our deployment descriptor files	179
19.	Separate HTTP server instances with WAS SE V3.5 and WebSphere for z/OS V4.0 plugins	181
20.	Sample VM user directory entry for our first Linux guest	195
21.	Overview of our Linux IP networking connectivity	196
22.	Sample VM user directory entry for TCP/IP.	197
23.	Portions of our PROFILE TCPIP on VM showing our networking interface definitions for our first Linux guest	198
24.	Sample PROFILE EXEC for our Linux virtual machine	199
25.	Overview of Linux networking with HiperSockets.	201
26.	Sample Linux VM user directory entry containing HiperSockets devices	203
27.	Example RMF Monitor I post processor summary report	209
28.	Example RMF Monitor III online sysplex summary report.	210
29.	Example RMF workload activity report in WLM goal mode	211

Tables

1.	Parallel Sysplex planning library publications	xix
2.	Our mainframe servers	5
3.	Our coupling facilities.	7
4.	Other sysplex hardware configuration details	10
5.	Our production OLTP application groups	13
6.	Summary of our workloads	16
7.	Our high-level migration process for z/OS V1R3	22
8.	Additional APARs for migrating to z/OS V1R3	23
9.	Our high-level migration process for z/OS.e V1R3	25
10.	Our tablespace allocation for our MQPRODDB database	34
11.	The tables and indexes for our MQPRODDB database	35
12.	GSS API environment variables we set in slapd.envvars	136
13.	Information about our IBM HTTP Server configuration.	157
14.	Information about our LDAP configuration	160
15.	Values we specified for IKYSETUP variables	160
16.	Values for IQD frame sizes and resulting TCP/IP MTU sizes	202
17.	Summary of our parmlib changes for z/OS V1R3 and z/OS.e V1R3.	207
18.	Available year-end editions of our test report	214
19.	Some IBM Web sites that we reference	215
20.	Other Web sites that we reference	216

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 quarterly 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.

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.

Introduction

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

- “Our Parallel Sysplex hardware configuration” on page 3
- “Our Parallel Sysplex software configuration” on page 12
- “Our workloads” on page 15
- “Our networking and application enablement configuration” on page 57

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 xix.

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 C, “Document availability and distribution” on page 213 for other ways to access our test reports.

We update and publish our test report on a quarterly basis. The quarterly editions are cumulative for the current year. At the end of each year, we freeze the content in our December edition; we begin with a new test report in March of the following year. The most recent quarterly edition as well as all of the previous year-end (December) editions are available on our Web site.

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 not familiar 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 MVS Setting Up a Sysplex</i>	SA22-7625
<i>z/OS Parallel Sysplex Overview: An Introduction to Data Sharing and Parallelism</i>	SA22-7661
<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/pso/
- 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 allows you to look up explanations for most of the z/OS, z/VM, and VSE messages you encounter, as well as system abends and some 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 access LookAt from the Internet at:

<http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/>

or from anywhere in z/OS where you can access a TSO command line (for example, TSO prompt, ISPF, z/OS UNIX System Services running OMVS). You can also download code from the *z/OS Collection* (SK3T-4269) and the LookAt Web site so you can access LookAt from a PalmPilot (Palm VIIx suggested).

To use LookAt on the Internet to find a message explanation, go to the LookAt Web site and simply enter the message identifier (for example, \$HASP701 or \$HASP*). You can select a specific release to narrow your search.

To use LookAt as a TSO command, you must have LookAt installed on your host system. You can obtain the LookAt code for TSO from a disk on your z/OS *Collection* (SK3T-4269) or from the LookAt Web site. To obtain the code from the LookAt Web site, do the following:

1. Go to <http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/>.

Introduction

2. Click **News**.
3. Scroll to **Download LookAt Code for TSO and z/VM**.
4. Click the ftp link, which will take you to a list of operating systems. Click the appropriate operating system. Then click the appropriate release.
5. Open the **lookat.me** file and follow its detailed instructions.

After you have LookAt installed, you can access a message explanation from a TSO command line by entering: **lookat** *message-id*. LookAt will display the message explanation for the message requested.

Note: Some messages have information in more than one book. For example, IEC192I can be found in *z/OS MVS System Messages, Vol 7 (IEB-IEE)* and also in *z/OS MVS Routing and Descriptor Codes*. For such messages, LookAt displays a list of books in which the message appears. You can then select one of the books to view the message explanation.

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 bixler@us.ibm.com 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

This topic summarizes the changes made to this document.

We update our test report quarterly with new information and experiences. If the edition you are reading is more than a few months old, check the IBM World Wide Web pages or consult your IBM representative to see if a newer edition is available (see “How to find the Parallel Sysplex Test Report” on page xviii).

Summary of changes for SA22-7663-05 June 2002

This document contains information previously presented in SA22-7663-04.

New information

- Implementing the DFSMSHsm common recall queue function
- Migrating to CICS TS V2.2
- Using access control lists (ACLs) in z/OS UNIX
- Miscellaneous enhancements to z/OS UNIX in z/OS V1R3
- Protecting IBM HTTP Server resources with LDAP using the SSL transport
- Our experiences with setting up and using PKI Services
- Our experiences with LDAP Server, including:
 - setting up referral processing between LDAP databases
 - enabling LDAP for Kerberos authentication
 - resolving a problem with corrupted DNSs

Changed information

- Our Parallel Sysplex hardware and software configuration
- Our section on migrating to z/OS.e V1R3 (added information to clarify the licensing implications and requirements)
- Enabling LDAP native authentication
- Our networking and application enablement configuration

Deleted information

- We removed most of the information in Chapter 6, “Parallel Sysplex automation” on page 51 in preparation for our coverage of msys for Operations. You can find information about our previous automations setup in our December 2001 edition.
- Information about how we set up our bookstore application. You can find the most recent information about our bookstore application in our December 2001 edition.
- Information about setting up Linux virtual images under S/390 Virtual Image Facility for LINUX (VIF). VIF is no longer available, but you can read about our previous experiences with it in our December 2001 edition.

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 SA22-7663-04
March 2002**

New information

- Migrating to z/OS V1R3
- Migrating to z/OS.e V1R3
- Enabling LDAP Server native authentication
- Setting up Security Server Network Authentication Service for z/OS

Changed information

- Our Parallel Sysplex hardware configuration, including the addition of an IBM @server zSeries 800 server
- Using Enhanced ASCII functionality
- Migrating to WebSphere Application Server V4.0.1 for z/OS and OS/390

Part 1. Parallel Sysplex

Chapter 1. About our Parallel Sysplex environment.	3
Overview of our Parallel Sysplex environment	3
Our Parallel Sysplex hardware configuration	3
Overview of our hardware configuration	3
Hardware configuration details.	5
Mainframe server details.	5
Coupling facility details	7
Other sysplex hardware details	10
Our Parallel Sysplex software configuration	12
Overview of our software configuration	12
About our naming conventions	14
Our VTAM configuration	14
Our workloads	15
Base system workloads.	16
Application enablement workloads.	17
Networking workloads	18
Database product workloads	19
Database product OLTP workloads	19
Database product batch workloads	20
MQSeries/DB2 bookstore application.	20
MQSeries workloads.	20
Chapter 2. Migrating to and using z/OS	21
Overview	21
Migrating to z/OS V1R3	21
z/OS V1R3 base migration experiences.	21
Our high-level migration process	21
More about our migration activities	22
Introducing z/OS.e V1R3	23
What is z/OS.e V1R3?	23
z/OS.e V1R3 base migration experiences	24
Our high-level migration process	24
More about our migration activities	26
Other experiences with z/OS.e V1R3.	28
Summary of requirements for z/OS.e V1R3	28
z/OS performance.	29
Chapter 3. Using MQSeries for OS/390 V5R2	31
Implementing shared queues.	31
Task 9: Setting up the DB2 environment for shared queues	33
Task 10: Setting up the coupling facility structures for shared queues	37
Task 12: Customizing the initialization input data sets for shared queues	38
Task 15: Adding the MQSeries entries to the DB2 data-sharing group.	39
Task 16: Tailor your system parameter module	40
Migrating our MQSeries-CICS bridge queues to shared queues	40
Migrating to a clustered MQSeries setup for SDSF.	41
Steps for migrating to a clustered MQSeries setup for SDSF	41
Chapter 4. Implementing the DFSMSHsm common recall queue	43
Steps for defining the DFSMSHsm common recall queue environment	43
Using the DFSMSHsm common recall queue	43
Starting CRQ processing	44
Displaying and controlling CRQ processing	44

Experiences with CRQ processing	44
Chapter 5. Migrating to CICS TS Version 2 Release 2	45
Overview of migrating to CICS TS 2.2	45
Performing the migration to CICS TS 2.2	46
Preparing for migration	46
Migrating CICSPlex SM.	47
Migrating the CASS	47
Steps for migrating the CASS	47
Migrating the CMASs	48
Steps for migrating the CMASs	48
Migrating the MASS	49
Steps for migrating the MASS	49
Experiences with migrating to CICS TS 2.2	49
Chapter 6. Parallel Sysplex automation	51

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:

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

The CPCs consist of the following machine types:

- Two IBM @server zSeries 900 (z900) processors
- One IBM @server zSeries 800 (z800) processor
- One IBM S/390 Parallel Enterprise Server, Generation 6 (G6)
- One IBM S/390 Parallel Enterprise Server, Generation 5 (G5)

The 14 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:
 - Two failure-independent coupling facilities that run in LPs on standalone CPCs
 - One non-failure-independent coupling facility that runs in an LP on a CPC that houses 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:

- One native Linux image
- Two z/VM images, each of which hosts multiple Linux virtual machines (LVMs)

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

Parallel Sysplex environment

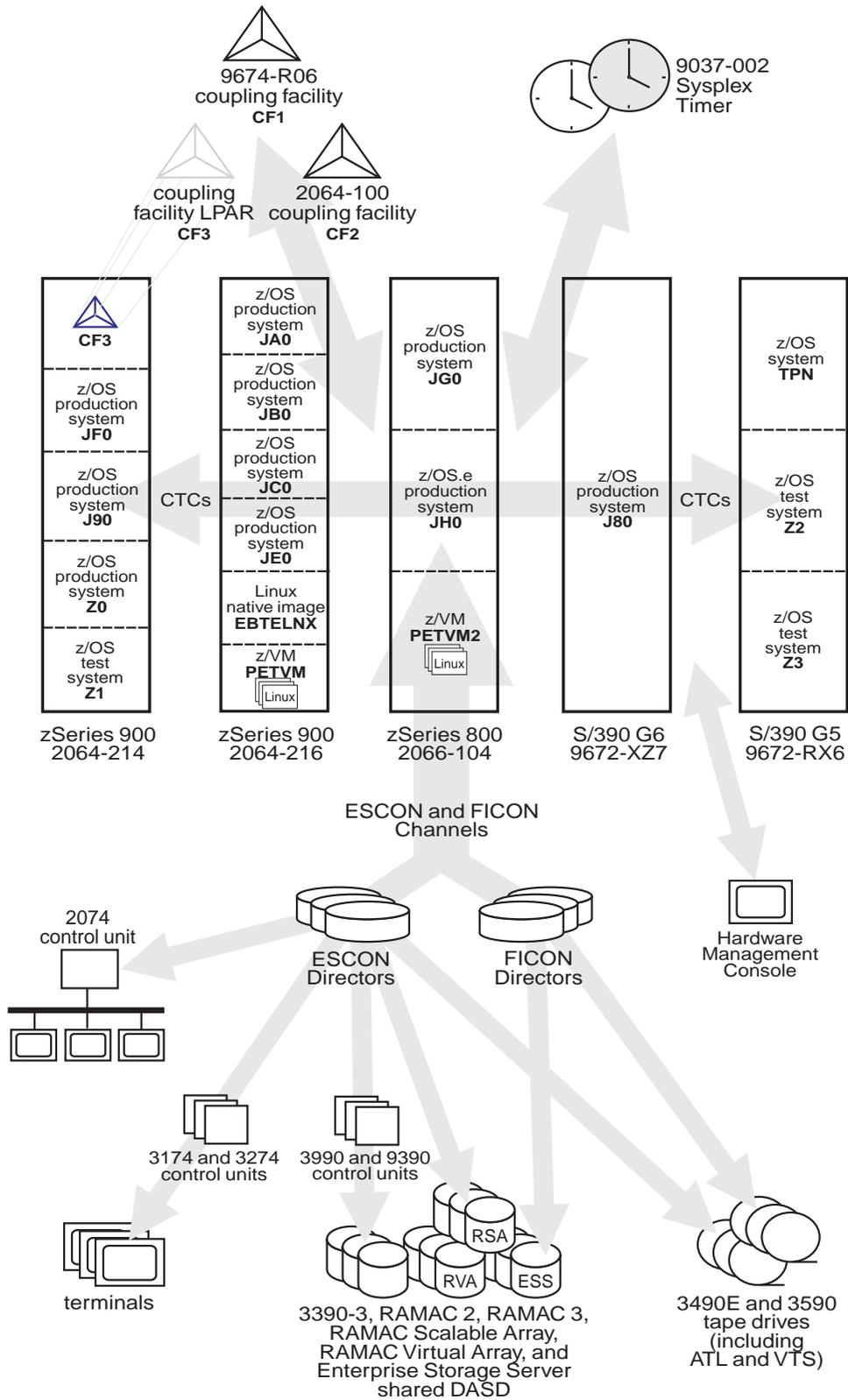


Figure 1. Our sysplex hardware configuration

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 1.

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 and CPs	Mode LPs	HSA	Storage: Central Expanded	System name — usage Virtual CPs (static, managed) Initial LPAR weight
IBM @server zSeries 800 Model 104 (2066-104) (see notes 4 and 5 below)	1 CPC	LPAR mode	176M	4000M	JG0 — z/OS production LP 4 shared CPs (2, 2) weight of 400
	4 CPs	3 LPs		3008M	JH0 — z/OS.e production LP 4 shared CPs weight of 400
				1024M	PETVM2 — z/VM LP 2 shared CPs weight of 199
IBM @server zSeries 900 Model 214 (2064-214) (see note 3 below)	1 CPC	LPAR mode	192M	4096M	JF0 — z/OS production LP 12 shared CPs (8, 4) weight of 285
	16 CPs	5 LPs (1 LP used as a coupling facility)		4096M	J90 — z/OS production LP 12 shared CPs (3, 9) weight of 285
				4096M	Z0 — z/OS production LP 12 shared CPs (7, 5) weight of 285
				4096M	Z1 — z/OS test LP 12 shared CPs (4, 8) weight of 145
IBM @server zSeries 900 Model 216 (2064-216) (see note 3 below)	1 CPC	LPAR mode	192M	7552M	JA0 — z/OS production LP 15 shared CPs (5, 10) weight of 250
	16 CPs	6 LPs		7168M	JB0 — z/OS production LP 15 shared CPs (4, 11) weight of 250
				7168M	JC0 — z/OS production LP 15 shared CPs (4, 11) weight of 250
				7552M	JE0 — z/OS production LP 15 shared CPs (10, 5) weight of 250
				256M	EBTELNX — native Linux image 1 shared CP
			1024M	PETVM — z/VM LP 2 shared CPs weight of 10	

Parallel Sysplex environment

Table 2. Our mainframe servers (continued)

Server model (Machine type-model)	CPCs and CPs	Mode LPs	HSA	Storage: Central Expanded	System name — usage Virtual CPs (static, managed) Initial LPAR weight
IBM S/390 Parallel Enterprise Server Generation 5 (G5) Model RX6 (9672-RX6)	1 CPC 10 CPs	LPAR mode 3 LPs	96M	2048M 1024M	TPN — z/OS LP for TPNS 8 shared CPs
				2048M 512M	Z2 — z/OS test LP 6 shared CPs
				2048M 400M	Z3 — z/OS test LP 6 shared CPs
IBM S/390 Parallel Enterprise Server Generation 6 (G6) Model XZ7 (9672-XZ7)	1 CPC 12 CPs	BASIC mode 0 LPs	64M	2048M 14336M	J80 — z/OS production LP 12 shared CPs

(see note 2 below)

Notes:

1. For the S/390 server models, evaluate whether you need to configure some of your storage as expanded storage based on the functions you are using. Some functions, such as Hiperbatch, require expanded storage.
2. For our G6, we applied the IZP version of IOCP 1.7.2, which is available with the fix for APAR OW36887 (PTF UW90547). We also applied the fix for HCD APAR OW37775 (PTFs UW90602, UW90603, and UW90604).
3. For our z900 servers, 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).
4. For our z800 server, we applied the fix for IOCP APAR OW52993. We also applied the fix for HCD APAR OW51339.
5. 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 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/.

Coupling facility details

Table 3 provides information about the coupling facilities in our sysplex. Figure 2 on page 10 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 Controlled by	Storage: Central Expanded	Channel distribution
CF1	S/390 9674-R06 Coupling Facility with expansion cage, feature code 2000 1 CPC with 7 CPs CFLEVEL=11 (CFCC Release 11.00, Service Level 01.06) Controlled by the HMC	2G 4G	<p>12 TYPE=CFR channels. These are intersystem coupling (ISC) channels (see note 1).</p> <p>There are 12 corresponding TYPE=CFS channels on the following systems ("shared" indicates that the systems share that number of channels using EMIF):</p> <ul style="list-style-type: none"> • JA0/JB0/JC0/JE0: 2 shared • JG0/JH0: 3 shared • TPN/Z2/Z3: 2 shared (HiPerLinks) • JF0/J90/Z0/Z1: 5 shared <hr/> <p>5 TYPE=CBR channels. These are integrated cluster bus (ICB) channels (see note 3).</p> <p>There are 5 corresponding TYPE=CBS channels on the following systems:</p> <ul style="list-style-type: none"> • JA0/JB0/JC0/JE0: 2 shared • J80: 2 shared • JF0/J90/Z0/Z1: 1 shared
CF2	IBM @server zSeries 900 Model 100 (2064-100) 1 CPC with 5 CPs CFLEVEL=12 (CFCC Release 12.00, Service Level 04.06) Controlled by the HMC	4G	<p>1 TYPE=CFR channel.</p> <p>There is a corresponding TYPE=CFS channel shared on systems TPN/Z2/Z3.</p> <hr/> <p>9 TYPE=CFP channels. These are a special type of ISC channels known as <i>peer</i> channels (see note 6).</p> <p>There are 9 corresponding TYPE=CFP channels on the following systems:</p> <ul style="list-style-type: none"> • JA0/JB0/JC0/JE0: 2 shared • JG0/JH0: 3 shared • JF0/J90/Z0/Z1: 4 shared <hr/> <p>5 TYPE=CBR channels.</p> <p>There are 5 corresponding TYPE=CBS channels on the following systems:</p> <ul style="list-style-type: none"> • J80: 3 shared • TPN/Z2/Z3: 2 shared <hr/> <p>4 TYPE=CBP channels. These are a special type of ICB channels known as <i>peer</i> channels (see note 6).</p> <p>There are 4 corresponding TYPE=CBP channels on the following systems:</p> <ul style="list-style-type: none"> • JA0/JB0/JC0/JE0: 2 shared • JF0/J90/Z0/Z1: 2 shared

Parallel Sysplex environment

Table 3. Our coupling facilities (continued)

Coupling facility name	Model CPCs and CPs CFLEVEL Controlled by	Storage: Central Expanded	Channel distribution
CF3	Coupling facility LP on a zSeries 900 Model 214 (2064-214) 2 dedicated CPs CFLEVEL=12 (CFCC Release 12.00, Service Level 04.06) Controlled by the HMC	4G	3 TYPE=CFR channels. (See note 2.) There are 3 corresponding TYPE=CFS channels on the following systems: <ul style="list-style-type: none"> • J80: 2 shared (HiPerLinks) • TPN/Z2/Z3: 1 shared (HiPerLink) <hr/> 3 TYPE=CBR channels. (See note 4.) There are 3 corresponding TYPE=CBS channels on the following systems: <ul style="list-style-type: none"> • J80: 1 shared • TPN/Z2/Z3: 2 shared <hr/> 5 TYPE=CBP channels. (See notes 4 and 6.) There are 5 corresponding TYPE=CBP channels on the following systems: <ul style="list-style-type: none"> • JA0/JB0/JC0/JE0: 2 shared • JG0/JH0: 2 shared • JF0/J90/Z0/Z1: 1 shared <hr/> 2 TYPE=ICP channels. These are a special type of internal coupling (IC) channel known as <i>peer</i> channels. (See notes 5 and 6.) There are 2 corresponding TYPE=ICP channels shared on systems JF0/J90/Z0/Z1.

Table 3. Our coupling facilities (continued)

Coupling facility name	Model	Storage: Central Expanded	Channel distribution
	CPCs and CPs CFLEVEL Controlled by		

Notes:

- All of the ISC channels on our S/390 (G5 and G6) server models are HiPerLinks.
 Note that the S/390 servers support two types of ISC channels: the standard 1K channels (with orange cables) and the 3K HiPerLink channels (with yellow cables). The z900 servers only support the 3K ISC channels in compatibility mode (versus peer mode); however, they are not termed "HiPerLinks" on these machines. We had to upgrade the port cards on our other servers where necessary to use the 3K ISC channels to connect them to our z900 coupling facilities.
- On our z900 server that contains CF3, the total number of ISC channels is 12:
 - Three are receiver channels (TYPE=CFR) and belong to the coupling facility LPAR (CF3).
 - Five are sender channels (TYPE=CFS) and are shared by the operating system LPARs to connect them to CF1 (see Figure 2).
 - Four are peer channels (TYPE=CFP) and are shared by the operating system LPARs to connect them to CF2 (see note 6).
- All of our servers also have coupling facility channels called integrated cluster bus (ICB) channels. These channels are faster than standard ISC channels and HiPerLinks. ICB channels are only available on the G5 and higher server models.
- On our z900 server that contains CF3, the total number of ICB channels is 12:
 - Three are receiver channels (TYPE=CBR) and belong to the coupling facility LPAR (CF3).
 - One is a sender channel (TYPE=CBS) and is shared by the operating system LPARs to connect them to CF1.
 - Eight are peer channels (TYPE=CBP), of which three are shared by the operating system LPARs to connect them to CF2 and CF3, and five connect CF3 to systems JA0/JB0/JC0/JE0, JG0/JH0, and JF0/J90/Z0/Z1 (see note 6).
- Our z900 server that contains CF3 also has internal coupling (IC) channels. IC channels are logical connections between a coupling facility partition and the operating system partitions 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 ISC (including HiPerLinks) and ICB channels.
- On the zSeries 800 and 900 servers, you can define coupling facility channels as peer channels on both sides of a coupling facility connection. A CF peer channel contains both sender and receiver functions; however, it is not required that these functions be used on both sides. 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 LPARs and operating system LPARs that reside on z800 and z900 servers. See *z/OS HCD Planning* for more information.

Figure 2 on page 10 illustrates our coupling facility channel configuration.

Parallel Sysplex environment

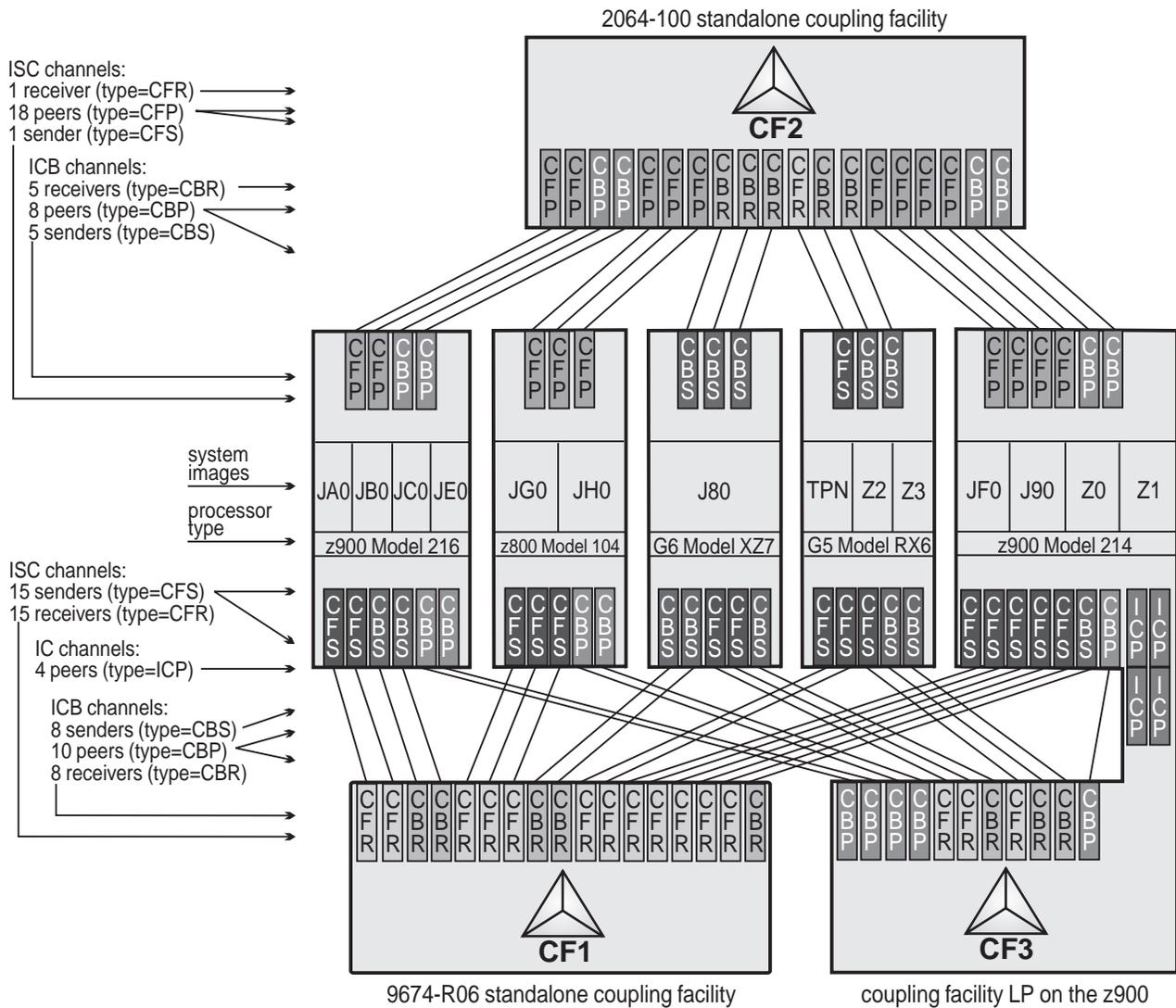


Figure 2. Our coupling facility channel configuration

Other sysplex hardware details

Table 4 highlights information about the other hardware components in our sysplex:

Table 4. 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 4. 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. Note: All systems 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 means of communication because it allows us to test more code paths, and 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, ICB, and IC channels	We have at least two paths between every system and each coupling facility. We use EMIF to logically share coupling channels among logical partitions.
	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 our G6, z800, and z900 servers 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. Note that one end of a FICON CTC (FCTC) connection must reside on a z800 server or a z900 server at driver level 3C or later.
Storage control units	3990 Storage Control Model 6 9390 (RAMAC 3) Storage Control	We have at least two paths to all DASDs. In fact, we have eight paths to all of our production workload databases.
DASD	3390 Model 3	All volumes shared by all systems; about 90% of our data is SMS-managed.
	RAMAC 2 (9392-002)	We recently added a RAMAC Virtual Array DASD. Like the RAMAC Scalable Array and the ESS, the RVA contains the control unit and storage media all in one box.
	RAMAC 3 (9392-003)	
	RAMAC Scalable Array (RSA, 9396-200)	We currently have three Enterprise Storage Servers; two are both ESCON- and FICON-attached (one is FICON-only to our z800) and the third is only ESCON-attached. Note: Although IBM recommends against running with both ESCON and FICON channel paths on an ESS except during the initial migration, we maintain both types of channels for greater flexibility in our testing.
	RAMAC Virtual Array (RVA, 9393)	
Enterprise Storage Server (ESS, 2105 Model F20)		
Tape	3490E tape drives	16 3490E tape drives that can be connected to any system.

Parallel Sysplex environment

Table 4. Other sysplex hardware configuration details (continued)

Hardware element	Model or type	Additional information
Automated tape library (ATL)	3495 Model L40 with 16 additional 3490E tape drives and 16 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 8 ESCON- and FICON-attached 3590 tape drives	All tape drives are accessible from all systems.

Our Parallel Sysplex software configuration

We run the z/OS V1R3 operating system along with CICS Transaction Server (CICS TS) V2R2, IMS V7R1 with its associated IRLM, DB2 UDB for z/OS and OS/390 V7 with its associated IRLM, and WebSphere Application Server V4.0 for z/OS and OS/390.

We also run z/OS.e V1R3 in one partition on our z800 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 V1R3 uses the same code as z/OS V1R3 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, IBM License Manager (ILM). ILM is a combination of license management tools that you use to manage licenses and check compliance with software terms and conditions. ILM is based on the industry standard for license management, XOpen Software License Management (XSLM).

ILM is not available for use at this time. ILM function is expected to be available at a future date. In the meantime, you must verify that the ILMMODE parameter in IEASYSxx is set to ILMODE=NONE. For information about ILM as it becomes available, see the ILM Web page at www.ibm.com/servers/eserver/zseries/zos/ilm/.

Overview of our software configuration

Figure 3 on page 13 shows a high-level view of our sysplex software configuration.

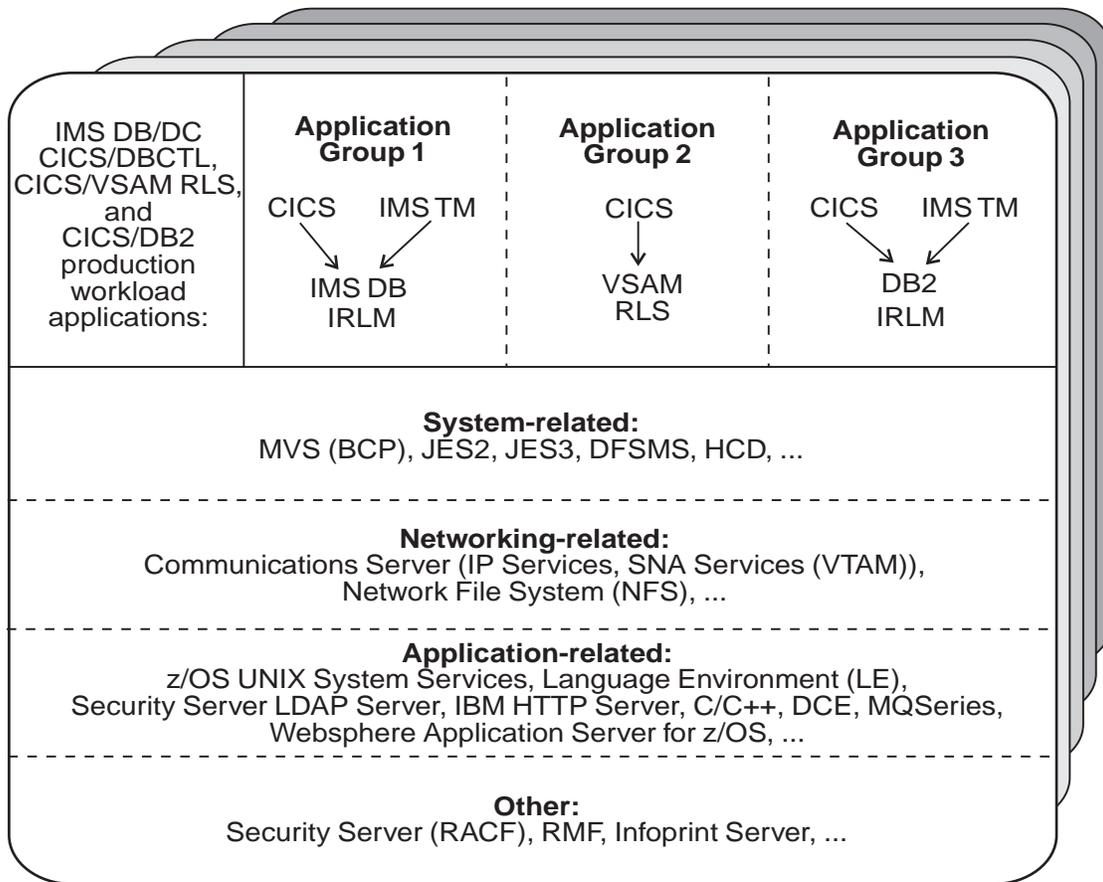


Figure 3. Our sysplex software configuration

We run three separate application groups in one sysplex and each application group spans multiple systems in the sysplex. Table 5 provides an overview of the types of transaction management, data management, and serialization management that each application group uses.

Table 5. 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 19.

Parallel Sysplex environment

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 jobnames 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 or T for TORs
- *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 VTAM configuration

Figure 4 on page 15 illustrates our current VTAM configuration.

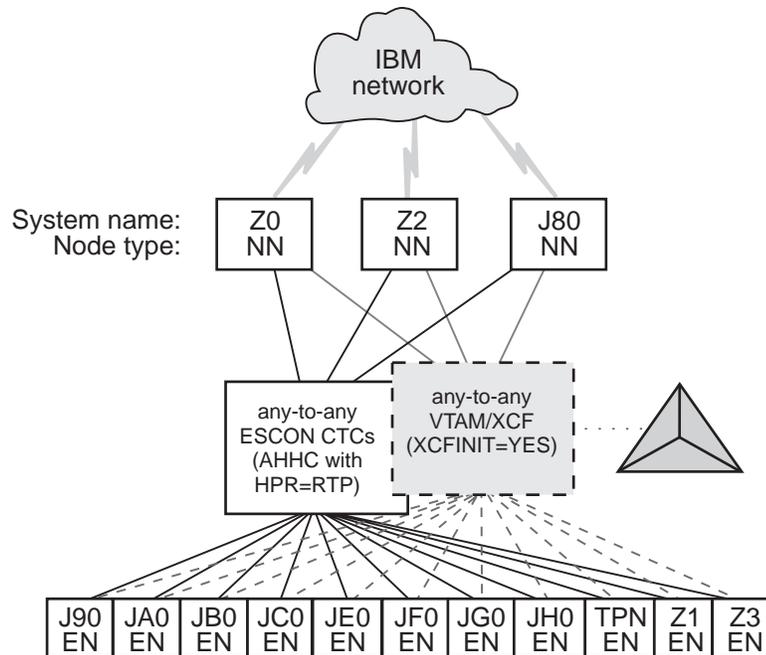


Figure 4. Our VTAM configuration

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 6 on page 16 summarizes the workloads we run during our prime shift and off shift. We describe each workload in more detail below.

Parallel Sysplex environment

Table 6. Summary of our workloads

Shift	Base system workloads	Application enablement workloads	Networking workloads	Database product workloads
Prime shift	<ul style="list-style-type: none"> Batch pipes Automatic tape switching JES2/JES3 printer simulators 	<ul style="list-style-type: none"> Shelltest (rlogin/telnet) Shelltest (TSO) UNIX recursive delete IBM HTTP Server WebSphere Application Server 	<ul style="list-style-type: none"> FTP workloads MMFACTS for NFS AutoWEB Silk Test NFS video stream TCP/IP CICS sockets TN3270 	<ul style="list-style-type: none"> CICS DBCTL IMS/DB2 IMS SMQ fast path IMS full function CICS/RLS online CICS/RLS batch CICS/DB2 CICS/QMF online queries DB2 online reorganization DB2/RRS stored procedure QMF batch queries DB2 Connect MQSeries/DB2 bookstore application MQSeries connection testing MQSeries batch stress MQSeries-CICS bridge workload
Off shift	<ul style="list-style-type: none"> Random batch Automatic tape switching JES2/JES3 printer simulators 	<ul style="list-style-type: none"> Shelltest (rlogin/telnet) Shelltest (TSO) UNIX recursive delete IBM HTTP Server WebSphere Application Server 	<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 RLS online CICS/DB2 QMF online queries CICS/RLS batch DB2 utility DB2 DDF IMS/DB2 MQSeries/DB2 bookstore application IMS utility

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 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:

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.

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.

UNIX recursive delete: This TPNS-driven workload copies over 700 directories from one large HFS to another. It then deletes all directories in the copy with one command.

IBM HTTP Server: These workloads are driven from both Windows NT and AIX/RISC workstations. They run against various HTTP server environments, including the following:

- HTTP scalable server
- HTTP standalone server
- HTTP standalone server running with a nonzero UID
- Sysplex distributor routing to various HTTP servers

These workloads access the following:

- DB2 through net.data
- MVS datasets
- FastCGI programs
- Counters
- Static html pages
- Protected pages

IBM WebSphere Application Server: This workload that simulates users making HTTP requests to drive Java servlets and JavaServer Pages (JSPs) which use various e-connectors to provide read and write access to our CICS/DB2, IMS-TM, and IMS-DB enterprise applications, and to our DB2 and VSAM data. A workstation-based tool generates the HTTP requests and verifies the results it receives. This workload uses the following connectors:

- JDBC, using a combination of direct DB2 calls and stored procedure calls, both with and without the following:
 - SSL
 - basic mode authentication (passwords in RACF)
 - connection pooling
 - session state
- SQLJ
- CICS Transaction Gateway (both hand-written programs and those created with VisualAge for Java)
- IMS Connector for Java (created with VisualAge for Java)
- Java Record I/O (to VSAM KSDS)
- CICS Web Support, using the Business Logic Interface plug-in to IBM HTTP Server with both EXCI access and 3270 Bridge access

Parallel Sysplex environment

- MQSeries Java classes
- XML Parser for Java
- Enterprise JavaBean (EJB) beans deployed on WebSphere Application Server Version 4.0
- JDBC, IMS, CICS connectors from Linux/390 using WebSphere Application Server Version 3.5 SE

Networking workloads

We run the following networking workloads:

FTP workloads:

- **FTP HFS/DB2:** This client/server workload simulates SQL/DB2 queries via an FTP client.
- **FTP HFS(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 WINDOWS 95/NT users 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.

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)
- 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

Highlights of our IMS data sharing workloads include:

- Using full function, fast path, and mixed mode transactions
- Accessing virtual storage option (VSO), shared sequential dependent (SDEP) databases, generic resources, and High Availability Large Databases (HALDB)
- Adding integrity checking on INSERT calls using SDEP journaling
- Adding a batch message processing (BMP) application to do integrity checking on REPLACE calls.

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

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.

Parallel Sysplex environment

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)
- DB2 batch workloads

We run our batch workloads under OPC 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 OPC, so that the jobs run in a certain sequence based on their inter-job dependencies.

MQSeries/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 MQSeries to bridge the platforms and MQSeries 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.

MQSeries workloads

We run three MQSeries workloads, all of them in batch:

Communications testing: We run a communications testing workload to test our channels. This workload kicks off an application that sends messages to a remote queue manager. A trigger monitor program running on the remote system then submits a batch job, which kicks off a separate application that sends the same messages back to the originating system.

MQSeries batch stress: This workload runs on one system and stresses MQSeries by issuing MQI calls. These calls include a variety of commands affecting queues of different types (temporary dynamic, permanent dynamic and alias). Parameters to the workload control the numbers of each type of call.

MQSeries-CICS bridge: Our MQSeries-CICS bridge workload uses the CICS bridge to run a transaction which updates a DB2 table. We changed the request and reply queues for our CICS bridge to be shared queues (see our December 2001 edition). For details on how we set up this workload, see our December 2000 edition.

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 V1R3”
- “Introducing z/OS.e V1R3” on page 23

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 53.

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 V1R1 and V1R2	our December 2001 edition
OS/390 V2R9 and V2R10	our December 2000 edition
OS/390 V2R7 and V2R8	our December 1999 edition
OS/390 V2R5 and V2R6	our December 1998 edition
OS/390 V1R3 and V2R4	our December 1997 edition
OS/390 V1R1 and V1R2	our December 1996 edition

Migrating to z/OS V1R3

This section describes our migration experiences with z/OS V1R3.

z/OS V1R3 base migration experiences

In this section we only describe our experiences with our base migration to z/OS V1R3, 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

The following is an overview of our z/OS V1R3 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 MVS Migration*, GA22-7580.

Note: Be sure to review *z/OS Program Directory*, GI10-0670. The program directory contains instructions to add data set SYS1.SDWDDLPA to the LPA. We missed this at first and it caused us some problems. This data set supports new function in DFSMS. See APAR II13131 for more information about this data set.

Table 7 shows the high-level process we followed to migrate the eligible members of our sysplex from z/OS V1R2 to z/OS V1R3.

Table 7. Our high-level migration process for z/OS V1R3

Stage	Description
Updating parmlib for z/OS V1R3	We created SYS1.PETR13.PARMLIB to contain all the parmlib members that changed for z/OS V1R3 and we used our LOADxx member for migrating our systems one at a time. (See “Using concatenated parmlib” 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.)
Updating the RACF templates	Before bringing up our first z/OS V1R3 system, we ran the IRRMIN00 utility with PARM=UPDATE to upgrade our existing RACF database templates to the current level that ships with z/OS V1R3 (see <i>z/OS Security Server RACF System Programmer's Guide</i> , SA22-7681 for details).
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. Also see “Additional APARs for z/OS V1R3” on page 23.
IPLing our first z/OS V1R3 image	We brought up z/OS V1R3 on two of our test systems (Z2 and Z3) and ran it there for about two weeks. (We kept one of our three test systems (Z1) at z/OS V1R2 for about four weeks while we used it to help debug a problem.)
IPLing additional z/OS V1R3 images	We continued to bring up additional z/OS V1R3 images across our sysplex, as follows: <ul style="list-style-type: none"> • We brought up z/OS V1R3 on three production systems (JE0, JF0, and JH0) and ran it on those systems for about one week. • We migrated three more production systems (JB0, JC0, and J80) to z/OS V1R3 and ran for about five days. During this time, we also migrated one of the z/OS V1R3 systems (JH0) on our z800 server to z/OS.e V1R3. See “Introducing z/OS.e V1R3” on page 23 for more information about this migration. • Next, we migrated system Z0 to z/OS V1R3 and ran for about two days. • We then migrated four more systems (JA0, JG0, J90, and Z1) to z/OS V1R3 and ran for about a week. • Finally, we migrated system TPN to z/OS V1R3.

The total time for our migration was approximately five weeks.

More about our migration activities

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 z/OS V1R2 and z/OS V1R3.

Using concatenated parmlib: We continue to use concatenated parmlib support to add or update parmlib members for z/OS R3. Appendix A, “Some of our parmlib

members” on page 207 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; not only does it isolate all the parmlib changes for z/OS V1R3 in one place, it also makes it easier to migrate multiple systems. Rather than change many parmlib members each time we migrate another system to V1R3, we just add the PARMLIB statements at the appropriate places in LOADxx to allow that system to use SYS1.PETR13.PARMLIB.

Data set names for JES2 and JES3 libraries: In z/OS V1R2, both JES2 and JES3 dropped the middle level qualifiers (MLQs) from their data set names. We followed suit at that time. Because neither JES2 nor JES3 changed in z/OS V1R3, we made no changes to their data set names for V1R3.

Recompiling REXX EXECs for automation: We had to recompile our SA OS/390 REXX EXECs when we migrated to z/OS V1R3. This is similar to what we did when we migrated to OS/390 R4, as we discuss in our December 1997 edition.

Additional APARs for z/OS V1R3: *z/OS and z/OS.e Planning for Installation* documents the compatibility APARs and PTFs that you need to prepare your systems for the migration to z/OS V1R3 (see the chapter on ensuring coexistence). We applied the coexistence service that the book prescribes. Table 8 lists additional APARs we want to highlight:

Table 8. Additional APARs for migrating to z/OS V1R3

Product / function	APAR number	Comments
IXFP (IBM Extended Facilities Product)	OW52933	Required for installations with RAMAC Virtual Array (RVA) DASD running IXFP

Introducing z/OS.e V1R3

This section introduces z/OS.e V1R3 and describes our experiences with migrating one z/OS image (JH0) from z/OS V1R3 to z/OS.e V1R3.

Note: Technically speaking, this is not *really* a migration since z/OS.e V1R3 is the *first* release of z/OS.e—there is no previous release of z/OS.e from which to migrate. However, we hope you’ll forgive us our loose terminology and simply understand that what we describe below is more of a “lateral move” from z/OS V1R3 to z/OS.e V1R3. If you’re not familiar with z/OS.e, the following brief overview will help clarify what we mean.

What is z/OS.e V1R3?

z/OS.e V1R3 is a specially priced offering of z/OS V1R3 that provides select function at an attractive price. z/OS.e is only available on the IBM @server zSeries 800 (z800) processor or a comparable non-IBM server. z/OS.e is intended to help customers exploit the fast growing world of next-generation e-business by making the deployment of new applications on the z800 very attractively priced.¹

z/OS.e is not a new operating system, nor is it a “stripped-down” or “lite” version of z/OS. Rather, z/OS.e V1R3 shares the same code base as z/OS V1R3, has the

1. z/OS.e is engine-licensed and customers must define the MSU capacity of a z/OS.e logical partition to be on an engine boundary. For details, see *z800 Software Pricing Configuration Technical Paper*, GM13-0121, available from the zSeries Library at www.ibm.com/servers/eserver/zseries/library/.

z/OS.e V1R3

same qualities of service, and adheres to the same release cycle and the same migration, fallback, service, and coexistence policies as z/OS V1R3. z/OS.e can also participate in a Parallel Sysplex—along with other z/OS or OS/390 members—and still retain its special pricing.

But, z/OS.e differs from z/OS in that it activates restrictions against running traditional workloads, such as CICS, IMS, COBOL, or FORTRAN applications (however, precompiled COBOL DB2 stored procedures and other precompiled COBOL applications using the Language Environment preinitialization interface, CEEPIPI, are supported). Also, selected z/OS base elements and optional features (as well as certain selected functions within those elements and features which support traditional workloads) are not licensed and not functional on z/OS.e.

z/OS.e is for new application workloads, such as those that are written in Java, Enterprise Java (J2EE), and C++, and that use WebSphere Application Server, DB2, MQSeries, Domino, and other enterprise applications (for instance, customer relationship management, supply chain management, enterprise resource planning, or business intelligence). The goal of z/OS.e is to offer excellent price performance for these workloads.

For more information about z/OS.e, including sample configurations, see:

- the z/OS.e home page at www.ibm.com/servers/eserver/zseries/zose/
- *z/OS.e Overview*, GA22-7869, available on the z/OS.e Internet library at www.ibm.com/servers/eserver/zseries/zose/bkserv/

z/OS.e V1R3 base migration experiences

In this section we only describe our experiences with our base migration to z/OS.e V1R3. It includes our high-level migration process along with other migration activities and considerations.

Our high-level migration process

The following is an overview of our z/OS.e V1R3 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 V1R3 and z/OS.e V1R3.

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 licence 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 one of our z/OS V1R3 systems to z/OS.e V1R3.

Table 9. Our high-level migration process for z/OS.e V1R3

Stage	Description
Obtaining licenses for z/OS.e	As described above, 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). Since our testing began before z/OS.e was orderable, we used 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 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. Therefore, we used HCD to change the name of the LPAR where z/OS.e will run to Z0SEJH0.
Updating parmlib for z/OS.e V1R3	z/OS.e requires the LICENSE=Z/0SE statement in the IEASYSxx parmlib member. We used the same SYS1.PETR13.PARMLIB data set that we created for z/OS V1R3. We then created separate IEASYSxx and IFAPRDxx members in SYS1.PARMLIB and tailored them specifically for z/OS.e V1R3. See "Updating system data sets for z/OS.e" on page 26 for details.
Updating our LOADxx member	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.

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 updated the entry for our system JH0 in our IEASYMPT member in SYS1.PETR13.PARMLIB to point to our new IFAPRD02 parmlib member and to reflect the new 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 V1R3. z/OS.e does not permit the use of a lower level JES2 (or JES3) or LE.

Updating our 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.

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 V1R3

Our testing of z/OS.e V1R3 included the following workloads or scenarios:

- z/OS UNIX System Services
- DB2 UDB V7
- IBM HTTP Server in scalable server mode
- WebSphere Application Server V3.5 SE and WebSphere for z/OS V4.0.1
- 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

Summary of requirements for z/OS.e V1R3

The following list summarizes the requirements for and limitations to running z/OS.e V1R3. However, this is not necessarily an exhaustive list. For complete information, see *z/OS and z/OS.e Planning for Installation* and *z/OS.e Licensed Program Specifications, GA22-7503*, available on the z/OS.e Internet library at www.ibm.com/servers/eserver/zseries/zose/bkserv/.

- z/OS.e V1R3 must run on an IBM @server zSeries 800 (z800) processor, machine type 2066, (or a comparable non-IBM server) in z/Architecture (64-bit) mode.
- z/OS.e V1R3 must run in LPAR mode in a logical partition named ZOSExxxx, where xxxx represents up to four permissible alphanumeric characters of your choice.

Software licensing for z/OS.e V1R3 is on a per-engine (CP) basis and you must define the MSU capacity of a z/OS.e logical partition to be on an engine boundary (see technical paper *z800 Software Pricing Configuration Technical Paper*). 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.

- You must specify LICENSE=Z/0SE in the active system parameter list (IEASYSxx) in parmlib or at the operator console during IPL.
- z/OS.e V1R3 supports a maximum of eight concurrent TSO/E sessions (including telnet sessions on port 23). However, telnet sessions on port 623 that directly login to the z/OS UNIX shell are not subject to this session limit.
- Like z/OS V1R3, z/OS.e V1R3 requires workload management (WLM) running in goal mode.
- You must use the current level of JES2 or JES3 that comes with z/OS.e; you cannot use a lower level of JES2 or JES3.
- You must use the current level of Language Environment (LE) that comes with z/OS.e; you cannot use a lower level of LE.

Language Environment Library Routine Retention (LRR) and Language Environment's use of the Run Time Library Services (RTL) are not functional and not licensed for use with z/OS.e.

Language Environment Compatibility Preinitialization for C and PL/I is not licensed for use with z/OS.e.

- z/OS.e will not execute CICS, IMS, COBOL, or FORTRAN applications. However, precompiled COBOL DB2 stored procedures and other precompiled applications that use the Language Environment preinitialization interface (CEEPIPI) are supported.

- You may not use the following compilers: COBOL, PL/I, VisualAge PL/I, and FORTRAN. However, z/OS.e supports execution of precompiled PL/I and VisualAge PL/I applications.
- The following elements and features of z/OS are not licensed for use with and are not functional in z/OS.e V1R3. You are not permitted to use these elements and features with z/OS.e V1R3:
 - BDT File-to-File
 - BookManager BUILD
 - BookManager READ
 - DCE Application Support
 - GDDM
 - GDDM-PGF
 - GDDM-REXX
 - LANRES
- The following elements of z/OS are not licensed for use with z/OS.e V1R3 and customers are not permitted to use these elements with z/OS.e V1R3:
 - Encina Toolkit Executive
 - MICR/OCR
- Customers cannot order or use the z/OS Communications Server Network Print Facility (NPF) optional feature with z/OS.e V1R3.
- The following DB2 features are not functional with z/OS.e V1R3. Customers are not permitted to use these features with z/OS.e V1R3:
 - DB2 QMF Host
 - DB2 QMF HPO

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 an internal response-time goal of 80% of CICS transactions completed in less than 0.6 seconds. We fill in the remaining 10% with batch work and various additional types of users, such as z/OS UNIX System Services (z/OS UNIX) users, 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 209.

z/OS performance

Chapter 3. Using MQSeries for OS/390 V5R2

IBM MQSeries is a family of products for cross-platform communication, using messages and queues. Using MQSeries products, 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. This is all explained in *MQSeries: An Introduction to Messaging and Queuing*, GC33-0805-01.

Currently, we're running with MQSeries for OS/390 V5R2. For information on how we migrated to this level see our December 2001 edition at:

http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/BOOKS/E0S2P603/CCONTENTS

In this chapter, we'll cover the following topics:

- "Implementing shared queues"
- "Migrating to a clustered MQSeries setup for SDSF" on page 41

Implementing shared queues

We set up shared queues in our MQSeries environment. A shared queue is a local queue that is shared by a group of queue managers. The group of queue managers that share a set of shared queues form the **queue sharing group**. Any of the queue managers in the queue sharing group can access messages on the shared queues. This means that within a queue sharing group, you can put a message on to a shared queue on one queue manager and a different queue manager can access the same message. This makes queue sharing a rapid way to communicate within the queue-sharing group, and eliminates the need for active channels between queue managers. Queue managers in the queue sharing group can all access the shared queues because the messages are stored in the coupling facility.

One difference between a queue sharing group and an MQSeries cluster is that the shared queues reside on the coupling facility. In a queue sharing group, the queue managers access the queues on the coupling facility. In our cluster, on the other hand, all the cluster queue managers can access the queues managed by all the other queue managers in the cluster, but the queues are on the individual system, rather than the coupling facility. (See our December 2001 edition for information about our bookstore application which we set up to use clustering.) Another difference is that clustering requires a set of channels on each queue manager in the cluster, but with shared queueing you don't need channels between the queue managers.

The shared queue definition is stored in a DB2 shared database called the shared repository. This means that we only have to define our shared queues once for all the queue managers in the queue-sharing group.

In a queue sharing group, an application can access any of the shared queues in the group simply by connecting to a local queue manager. Connectivity does not depend on the availability of a specific queue manager; any queue manager in the queue-sharing group can service the queue.

In order to implement shared queues, the following are required:

- Run with a system at the OS/390 V2R9 level or higher
- Have a coupling facility with level 9 or higher Coupling Facility Control Code (CFCC)

MQSeries

- DB2 and RRS on each system where MQSeries is running. Note that when we did this testing, our DB2 systems were still at the Version 5 level. (We subsequently migrated to DB2 V7; see our December 2001 edition.

Note that persistent messages are not supported with queue sharing. The maximum message length allowed on a shared queue is 63KB.

Our queue sharing environment looks as follows:

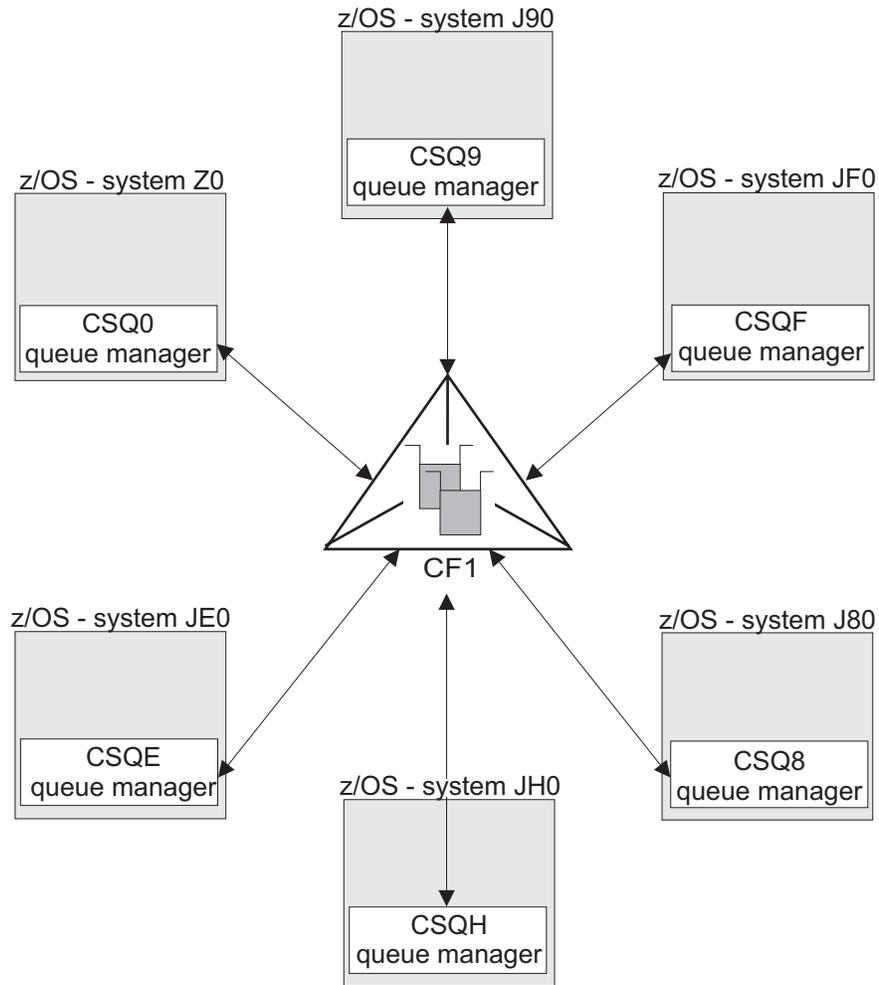


Figure 5. Our shared queue configuration

The shared queues are from our MQSeries-CICS bridge application, the CICS bridge request and reply queues. See our December 2000 edition for details on how we set up our pre-shared queue MQSeries-CICS bridge environment.

We used the following books to set up our shared queue configuration:

- *MQSeries for OS/390 Concepts and Planning Guide*, GC34–5650
- *MQSeries for OS/390 System Setup Guide*, SC34–5651
- *DB2 for OS/390 V6 Installation*, GC26–9008

Most of the setup for shared queue went smoothly. We have the following experiences to share about a few of the tasks in the chapter on customizing queue managers in *MQSeries for OS/390 System Setup Guide*:

- “Task 9: Setting up the DB2 environment for shared queues” on page 33

- “Task 10: Setting up the coupling facility structures for shared queues” on page 37
- “Task 12: Customizing the initialization input data sets for shared queues” on page 38
- “Task 15: Adding the MQSeries entries to the DB2 data-sharing group” on page 39
- “Task 16: Tailor your system parameter module” on page 40
- “Migrating our MQSeries-CICS bridge queues to shared queues” on page 40

You can find all the sample JCL mentioned in these experiences in data set hlq.SCSQPROC.

Task 9: Setting up the DB2 environment for shared queues

We have the following experiences to share in following Task 9, setting up the DB2 environment for shared queues, in *MQSeries for OS/390 System Setup Guide*:

Customizing the CSQ45CSG sample JCL to create the storage group: The CSQ45CSG sample JCL creates the storage group for the MQSeries database, tablespaces, and tables for our shared queue environment. All of the jobs supplied in CSQ45CSG run as batch jobs using the DB2-supplied program, DSNTIAD, and execute DB2 commands under TSO. The example below shows a portion of the CSQ45CSG JCL:

```
//SYSTSIN DD *
      DSN SYSTEM(DB1G)
      RUN PROGRAM(DSNTIAD) -
          PLAN(DSNTIA61) -
          LIB('DB2.DB2610.RUNLIB.LOAD')
/*
//SYSIN DD *
      CREATE STOGROUP "MQSTG"
          VOLUMES('*') VCAT MQSDB2;
/*
```

We coded the JCL statements as follows:

- The value for DSN should be the batch group attachment name that we use to access the DB2 data sharing group. DB1G is the DB2 subsystem ID and batch group attachment name. To find the batch group attachment name, we use the DB2 DIS GROUP command:

```
@dbz1 dis group
      DBZ1MSTR DSN7100I @DBZ1 DSN7GCMD
      *** BEGIN DISPLAY OF GROUP(DSNDB1G ) GROUP LEVEL(610)
                                          GROUP ATTACH NAME(DB1G)
-----
```

- In the RUN statement, DSNTIAD is the DB2-supplied batch program, DSNTIA61 is our plan name for the program, and DB2.DB2610.RUNLIB.LOAD is the name of the data set where DSNTIAD resides in our environment.
- The SYSIN DD statement executes the CREATE STOGROUP DB2 command:
- In the CREATE STOGROUP statement, MQSTG is the name of the DB2 storage group we're creating for our database, and MQSDB2 is the high level qualifier for the ICF catalog of the storage group.

Customizing the CSQ45CDB sample JCL to create the database for the queue sharing group: The CSQ45CDB sample JCL creates the database used by all the queue managers in the queue sharing group, which is also the DB2 datasharing group. The example below shows our input to the CSQ45CDB job (for the complete JCL see the sample in the MQSeries SCSQPROC dataset):

MQSeries

```
//SYSTSIN DD *
      DSN SYSTEM(DB1G)
      RUN PROGRAM(DSNTIAD) -
          PLAN(DSNTIA61) -
          LIB('DB2.DB2610.RUNLIB.LOAD')
/*
//SYSIN DD *
      CREATE DATABASE "MQPRODDB"
      BUFFERPOOL BP32K1
      STOGROUP MQSTG ;
/*
```

- **MQPRODDB** is the name of DB2 database we're creating.
- **BP32K1** is the name of the 32K DB2 buffer pool associated with the MQPRODDB database. We obtained this bufferpool name from our DB2 administrator.
- **MQSTG** is the name of the DB2 storage group associated with the MQPRODDB database.

Customizing the CSQ45CTS sample JCL to create the tablespaces for our queue sharing group: The CSQ45CTS sample JCL creates the tablespaces that contain the queue manager and channel initiator tables for our queue sharing group. Note all our tablespace names begin with 'TS' — this is our naming convention. The following table shows our tablespace allocation for our MQPRODDB database:

Table 10. Our tablespace allocation for our MQPRODDB database

Tablespace name	DB2 storage group	PRIQTY value	SECQTY value	4K bufferpool
TSADMIN	MQSTG	8000	1000	BP3
TSNAMES	MQSTG	1550	200	BP32K1
TSCHIN	MQSTG	55	10	BP3

To determine the value for PRIQTY (primary space allocation) for each tablespace, we used the table in *MQSeries for OS/390 Concepts and Planning Guide* to estimate how many rows of each table we expected to have. Then, we used the following formula to determine how many kilobytes we needed for our tables:

$$(\text{number of rows} * \text{length of rows}) / 1000 = \text{number of kilobytes needed}$$

Finally, we summed up all the values in kilobytes for all the tables in the same tablespace. We based our values for SECQTY (the secondary space allocation) on the value used for PRIQTY.

The following example shows the SYSTSIN and SYSIN statements of our CSQ45CTS JCL:

```
//SYSTSIN DD *
      DSN SYSTEM(DB1G)
      RUN PROGRAM(DSNTIAD) -
          PLAN(DSNTIA61) -
          LIB('DB2.DB2610.RUNLIB.LOAD')
/*
//SYSIN DD *
      CREATE TABLESPACE "TSADMIN"
      USING STOGROUP MQSTG
      PRIQTY 8000
      SECQTY 1000
      PCTFREE 20
      SEGSIZE 64
      BUFFERPOOL BP3
      LOCKSIZE ANY
      CLOSE NO
      IN MQPRODDB;
```

```

CREATE TABLESPACE "TSNAMES"
USING STOGROUP MQSTG
PRIQTY 1550
SECQTY 200
BUFFERPOOL BP32K1
LOCKSIZE ANY
CLOSE NO
IN MQPRODDB;

CREATE TABLESPACE "TSCHIN"
USING STOGROUP MQSTG
PRIQTY 55
SECQTY 10
FREEPAGE 10
PCTFREE 30
SEGSIZE 64
BUFFERPOOL BP3
LOCKSIZE ANY
CLOSE NO
IN MQPRODDB;
/*

```

Customizing the CSQ45CTB sample JCL to create DB2 tables and indexes for our queue sharing group: We used the CSQ45CTB JCL to create the following tables and indexes for our tablespaces for our queue sharing group:

Table 11. The tables and indexes for our MQPRODDB database

Tablespace	Table or index name	PRIQTY value (in KB)	SECQTY value (in KB)
TSADMIN	CSQ.ADMIN_QSG	5	2
	CSQ.ADMIN_QMGR	15	5
	CSQ.ADMIN_STRUCTURE	20	5
	CSQ.ADMIN_SCST	20	5
	CSQ.OBJ_QUEUE	4000	100
	CSQ.OBJ_PROCESS	250	50
	CSQ.OBJ_STGCLASS	10	2
TSNAMES	CSQ.OBJ_B_NAMELIST	1000	200
	CSQ.ADMIN_SCST_IX1	30	10
	CSQ.ADMIN_SCST_IX2	30	10
	CSQ.ADMIN_SCST_IX3	30	10
	CSQ.ADMIN_SCST_IX4	30	10
	CSQ.ADMIN_SCST_IX5	30	10
TSCHIN	CSQ.OBJ_B_CHANNEL	2000	200
	CSQ.ADMIN_SSKT_IX1	20	5

- **TSADMIN** is the tablespace used for the administration tables.
- **TSNAMES** is the tablespace used by the CSQ.OBJ_B_NAMELIST table.
- **TSCHIN** is the tablespace used for the channel initiator table.

Customizing the CSQ45BPL sample JCL to bind the DB2 plans: We used the CSQ45BPL sample JCL to bind the DB2 plans for the queue managers, utilities, and channel initiator. Our customized JCL looks as follows:

```

//SYSTSIN DD *
DSN SYSTEM(DB1G)

BIND PLAN(CSQ5D220) -

```

MQSeries

```
MEM(CSQ5D220) -
ACQUIRE(USE) RELEASE(COMMIT) -
CURRENTDATA(NO) -
ACT(REP) RETAIN ISOLATION(CS) -
LIB('MQS.V5R2M0.SCSQDEFS')

BIND PLAN(CSQ5L220) -
MEM(CSQ5L220) -
ACQUIRE(USE) RELEASE(COMMIT) -
CURRENTDATA(NO) -
ACT(REP) RETAIN ISOLATION(CS) -
LIB('MQS.V5R2M0.SCSQDEFS')

BIND PLAN(CSQ5R220) -
MEM(CSQ5R220) -
ACQUIRE(USE) RELEASE(COMMIT) -
CURRENTDATA(NO) -
ACT(REP) RETAIN ISOLATION(CS) -
LIB('MQS.V5R2M0.SCSQDEFS')

BIND PLAN(CSQ5U220) -
MEM(CSQ5U220) -
ACQUIRE(USE) RELEASE(COMMIT) -
CURRENTDATA(NO) -
ACT(REP) RETAIN ISOLATION(CS) -
LIB('MQS.V5R2M0.SCSQDEFS')

BIND PLAN(CSQ5W220) -
MEM(CSQ5W220) -
ACQUIRE(USE) RELEASE(COMMIT) -
CURRENTDATA(NO) -
ACT(REP) RETAIN ISOLATION(CS) -
LIB('MQS.V5R2M0.SCSQDEFS')

BIND PLAN(CSQ5B220) -
MEM(CSQ5B220) -
ACQUIRE(USE) RELEASE(COMMIT) -
CURRENTDATA(NO) -
ACT(REP) RETAIN ISOLATION(CS) -
LIB('MQS.V5R2M0.SCSQDEFS')

BIND PLAN(CSQ52220) -
MEM(CSQ52220) -
ACQUIRE(USE) RELEASE(COMMIT) -
CURRENTDATA(NO) -
ACT(REP) RETAIN ISOLATION(CS) -
LIB('MQS.V5R2M0.SCSQDEFS')
BIND PLAN(CSQ5S220) -
MEM(CSQ5S220) -
ACQUIRE(USE) RELEASE(COMMIT) -
CURRENTDATA(NO) -
ACT(REP) RETAIN ISOLATION(CS) -
LIB('MQS.V5R2M0.SCSQDEFS')

BIND PLAN(CSQ5K220) -
MEM(CSQ5K220) -
ACQUIRE(USE) RELEASE(COMMIT) -
CURRENTDATA(NO) -
ACT(REP) RETAIN ISOLATION(CS) -
LIB('MQS.V5R2M0.SCSQDEFS')
```

/*

Customizing the CSQ45GEX sample JCL to grant execute authority to the plans: We used the CSQ45GEX sample JCL to grant execute authority to the plans for the user IDs for use by the queue manager, utilities, and channel initiator. Our CSQ45GEX JCL looks as follows:

```
//SYSTSIN DD *
  DSN SYSTEM(DB1G)
  RUN PROGRAM(DSNTIAD) -
    PLAN(DSNTIA61) -
```

```

LIB('DB2.DB2610.RUNLIB.LOAD')
/*
//SYSIN DD *
GRANT EXECUTE ON PLAN CSQ5D220 TO qmgruser ;
GRANT EXECUTE ON PLAN CSQ5L220 TO qmgruser;
GRANT EXECUTE ON PLAN CSQ5R220 TO qmgruser;
GRANT EXECUTE ON PLAN CSQ5U220 TO qmgruser;
GRANT EXECUTE ON PLAN CSQ5W220 TO qmgruser;

GRANT EXECUTE ON PLAN CSQ5S220 TO qmgruser;
GRANT EXECUTE ON PLAN CSQ5S220 TO qmgruser;
GRANT EXECUTE ON PLAN CSQ5K220 TO qmgruser;

GRANT EXECUTE ON PLAN CSQ5B220 TO pgsguser ;
GRANT EXECUTE ON PLAN CSQ52220 TO utiluser ;
/*


- qmgruser is a user ID used for a queue manager started task
- pgsguser is the user ID used for the CSQ5PQSG utility
- utiluser is the user ID used for the CSQUTIL utility

```

Task 10: Setting up the coupling facility structures for shared queues

We have the following experiences to share in following Task 10, setting up the coupling facility structures for shared queues in *MQSeries for OS/390 System Setup Guide*:

Requirement for coupling facility structures for shared queues: In order to implement a shared queue environment, the MQSeries structures must reside on a coupling facility with level 9 or higher Coupling Facility Control Code (CFCC).

Updating the CFRM policy: We updated our CFRM policy to define the MQSeries administration coupling facility structure and the message queue structure.

MQSeries has the following naming requirements for the structures:

- Both structures must start with the queue sharing group name.
- MQSeries mandates that the name of the MQSeries administration structure be *queue_sharing_group*CSQ_ADMIN. For example, our queue sharing group name is MQGP, so MQSeries names our administration queue structure **MQGPCSQ_ADMIN**.
- MQSeries mandates that the name for the message queue structure be 12 characters or less in length and then have the queue sharing group name appended to the front. This allows you a total of 16 characters specified in the CFRM policy. We called our message queue structure **MQGPMSGQ1**.

We updated our CFRM policy as follows for our two queue sharing structures:

```

STRUCTURE NAME( MQGPCSQ_ADMIN )
    INITSIZE(10000)
    SIZE(16000)
    PREFLIST(CF3,CF2)
    REBUILDPERCENT(1)
    FULLTHRESHOLD(85)

STRUCTURE NAME( MQGPMSGQ1 )
    INITSIZE(16000)
    SIZE(32000)
    PREFLIST(CF2,CF3)
    REBUILDPERCENT(1)
    FULLTHRESHOLD(85)

```

We used the chapter on planning the coupling facility and DB2 environment in *MQSeries for OS/390 Concepts and Planning Guide* to calculate the sizes for our coupling facility structures.

Task 12: Customizing the initialization input data sets for shared queues

Each MQSeries instance gets its initial definitions from a series of commands contained in the MQSeries initialization input data sets. These data sets are referenced by the DDnames CSQINP1 and CSQINP2 defined in the MQSeries subsystem started task procedure. Sample initialization input data sets are supplied with MQSeries, see the *MQSeries for OS/390 Concepts and Planning Guide* for details.

Customizing the CSQ4INSS sample JCL to define the shared queue: The CSQ4INSS sample JCL contains system definitions for our shared queue. We customized the **CFSTRUCT** value in CSQ4INSS to specify the structure where our shared queue resides. Note that the value for CFSTRUCT is **not** the same as the structure name we defined in the CFRM policy. CFSTRUCT is actually the CFRM structure name minus the queue sharing group name. For example, in the CFRM policy, we defined the name of the message queue structure as MQGPMSGQ1. In the CFSTRUCT parameter of the queue definition, we specify MSGQ1 as the structure name. This had us stumped for a while until we figured out that the CFRM structure name and CFSTRUCT are specified differently. The distinction is designed to ensure isolation between queue sharing groups and to allow migration from test to production queue sharing environments without MQSeries definition changes.

After we customized CSQ4INSS, we added it to the CSQINP2 concatenation of our MQSeries started task procedure.

Our customized CQ4INSS JCL is as follows:

```
DEFINE QLOCAL( 'SYSTEM.QSG.CHANNEL.SYNCQ' ) +
        QSGDISP( SHARED ) +

    * Common queue attributes
        DESCR( 'System shared channel synchronization queue' ) +
        PUT( ENABLED ) +
        DEFPRTY( 5 ) +
        DEFPSIST( NO ) +
        CLUSTER( ' ' ) CLUSNL( ' ' ) DEFBIND( OPEN ) +

    * Local queue attributes
        GET( ENABLED ) +
        SHARE +
        DEFSOPT( SHARED ) +
        MSGDLVSQ( FIFO ) +
        RETINTVL( 999999999 ) +
        MAXDEPTH( 999999999 ) +
        MAXMSGL( 64512 ) +
        NOHARDENBO +
        BOTHRESH( 0 ) +
        BOQNAME( ' ' ) +
        STGCLASS( ' ' ) +
        INDXTYPE( MSGID ) +
        CFSTRUCT( 'MSGQ1' ) +

    * Event control attributes
        QDPMAXEV( ENABLED ) +
        QDPHIEV( DISABLED ) +
        QDEPTHHI( 80 ) +
        QDPLOEV( DISABLED ) +
        QDEPTHLO( 40 ) +
        QSVCEV( NONE ) +
        QSVCI( 999999999 ) +

    * Trigger attributes
        NOTRIGGER +
        TRIGTYPE( NONE ) +
```

```

        TRIGMPRI( 0 ) +
        TRIGDPTH( 1 ) +
        TRIGDATA( ' ' ) +
        PROCESS( ' ' ) +
        INITQ( ' ' )
*
*
*****
DEFINE QLOCAL( 'SYSTEM.QSG.TRANSMIT.QUEUE' ) +
        QSGDISP( SHARED ) +

* Common queue attributes
DESCR( 'System group transmission queue' ) +
PUT( ENABLED ) +
DEFPRTY( 5 ) +
DEFPSIST( NO ) +
CLUSTER( ' ' ) CLUSNL( ' ' ) DEFBIND( OPEN ) +

* Local queue attributes
GET( ENABLED ) +
SHARE +
DEFSOPT( SHARED ) +
MSGDLVSQ( FIFO ) +
RETINTVL( 999999999 ) +
MAXDEPTH( 999999999 ) +
MAXMSGL( 64512 ) +
NOHARDENBO +
BOTHRESH( 0 ) +
BOQNAME( ' ' ) +
STGCLASS( ' ' ) +
USAGE( XMITQ ) +
INDXTYPE( CORRELID ) +
CFSTRUCT( 'MSGQ1' ) +

* Event control attributes
QDPMAEV( ENABLED ) +
QDPHIEV( DISABLED ) +
QDEPTHHI( 80 ) +
QDPLOEV( DISABLED ) +
QDEPTHLO( 40 ) +
QSVIEV( NONE ) +
QSVICINT( 999999999 ) +

* Trigger attributes
NOTRIGGER +
TRIGTYPE( NONE ) +
TRIGMPRI( 0 ) +
TRIGDPTH( 1 ) +
TRIGDATA( ' ' ) +
PROCESS( ' ' ) +
INITQ( ' ' )

```

Task 15: Adding the MQSeries entries to the DB2 data-sharing group

We have the following experiences to share about adding the MQSeries entries to the DB2 data-sharing group:

Customized the CSQ45AQS sample JCL to add a queue sharing group entry to the MQSeries DB2 tables: We customized the CSQ45AQS sample JCL to add a queue sharing group entry to the MQSeries DB2 tables using the ADD QSG function of the CSQPQSG program. Our customized CSQ45AQS JCL is as follows:

```

//ADDQSG EXEC PGM=CSQ5PQSG,REGION=4M,
//          PARM='ADD QSG, MQGP, DSNDB1G, DB1G '

```

The parameters we customized were:

- **MQGP** is the queue sharing group name we're adding to the DB2 tables.
- **DSNDB1G** is the name of the DB2 data-sharing group that the queue sharing group uses.

MQSeries

- **DB1G** is the DB2 subsystem ID or batch group attach name used to access the DB2 data-sharing group

Customized the CSQ45AQM sample JCL to add entries for each queue

manager: We customized the CSQ45AQM JCL that adds an entry for each queue manager in the queue sharing group to the MQSeries DB2 tables. Our customized CSQ45AQM is as follows:

```
//ADDQMGR8 EXEC PGM=CSQ5PQSG,REGION=4M,  
// PARM='ADD QMGR,CSQ8,MQGP,DSNDB1G,DB1G'
```

We added the same parameters that we did for the CSQ45AQS JCL sample.

Task 16: Tailor your system parameter module

The MQSeries system parameter module controls the logging, archiving, tracing, and connection environments that MQSeries uses in its operation. We had to customize the CSQ6SYSP macro, which specifies the connection and tracing parameters. We added the following line to the CSQ6SYSP macro for each MQSeries instance in the queue sharing group:

```
"QSGDATA=(MQGP,DSNDB1G,DB1G,4),"
```

The parameters we added were:

- MQGP is the queue sharing group name.
- DSNDB1G is the name of the DB2 data-sharing group.
- DB1G is the DB2 group attach name.
- 4 is the number of server tasks accessing DB2. (4 is the default.)

Migrating our MQSeries-CICS bridge queues to shared queues

When we set up our shared queue environment, we decided to migrate our MQSeries-CICS bridge request and reply queue to a shared one. (See our December 2000 edition for details about our pre-shared queue MQSeries-CICS bridge application.) The example below shows how we changed the MQSeries definition for one of our queues, the reply queue, to make it a shared queue:

```
*****  
* Definitions for CICS Bridge/Adapter *  
*****  
DEFINE QLOCAL( 'CICS3A0A.BRIDGE.REPLY' ) REPLACE +  
  DESCR( 'MQ reply to queue' ) +  
  PUT( ENABLED ) +  
  DEFPRTY( 0 ) +  
  DEFPSIST( YES ) +  
  GET( ENABLED ) +  
  SHARE +  
  INDXTYPE(CORRELID) +  
  CFSTRUCT(MSGQ1) +  
  MSGDLVSQ( PRIORITY ) +  
  RETINTVL( 99999999 ) +  
  MAXDEPTH( 99999999 ) +  
  MAXMSGL( 4194304 ) +  
  NOHARDENBO +  
  BOTHRESH( 0 ) +  
  BOQNAME( ' ' ) +  
  STGCLASS( 'CICSSTG' ) +  
  USAGE( NORMAL ) +  
  QDPMAXEV( ENABLED ) +  
  QDPHIEV( DISABLED ) +  
  QDEPTHHI( 80 ) +  
  QDPLOEV( DISABLED ) +  
  QDEPTHLO( 40 ) +  
  QSVCIIEV( NONE ) +  
  QSVCIINT( 99999999 ) +  
  NOTRIGGER +
```

```

TRIGTYPE( FIRST ) +
TRIGDPTH( 1 ) +
TRIGMPRI( 0 ) +
TRIGDATA( ' ' ) +
PROCESS( ' ' ) +
INITQ( ' ' )
*
*****
***** CICS BRIDGE DEFINITIONS *****
*****
DEFINE QLOCAL('CICS3A0A.BRIDGE.QUEUE') REPLACE +
  DESCR('CICS BRIDGE REQUEST QUEUE') +
  MSGDLVSQ(FIFO) +
  DEFPSIST(YES) +
  SHARE +
  CFSTRUCT(MSGQ1) +
  DEFPSIST(YES) +
  HARDENBO +
  TRIGGER TRIGTYPE(FIRST) +
  PROCESS('CICS_BRIDGE') +
  INITQ('CICS3A0A.INITQ')

```

We made the following changes:

- **REPLACE +** : We specify **REPLACE +** to replace the values in the bridge reply queue definition with our new ones.
- **SHARE +** : We specify **SHARE +** to indicate that the queue is a shared one.
- **CFSTRUCT(MSGQ1)** : We specify the structure where the shared queue resides. Note that the value for CFSTRUCT is name of the structure specified in the CFRM **minus** the queue sharing group name.

Migrating to a clustered MQSeries setup for SDSF

SDSF uses MQSeries for communication between SDSF servers in a sysplex. Up until recently, we've had queue manager aliases and channels defined between each system in our sysplex for SDSF. Because we have 16 queue managers in our configuration now, this setup requires that we define 256 channels, one from each queue manager to each remote queue manager. SDSF has been enhanced to make use of MQSeries clusters which significantly reduce the number of MQ channels required. For more information, see the following books:

- *MQSeries Queue Manager Clusters*
- *z/OS SDSF Operation and Customization*

Steps for migrating to a clustered MQSeries setup for SDSF

Below are the steps we took to migrate from a non-clustered MQ setup for SDSF to a clustered one. Note that we've named our new cluster SDSF.CLUSTER.

1. Defined cluster sender and receiver channels on each system; receiver channels for other systems to access this queue manager and sender channels for this queue manager to access a repository.

For example, the sender and receiver channel definitions for one system look as follows:

```

DEFINE CHANNEL(SDSF.TO.J80) CHLTYPE(CLUSRCVR) +
DESCR('CLUSTER RECEIVER CHANNEL') +
TRPTYPE(TCP) CONNAME(ip_address) +
CLUSTER(SDSF.CLUSTER) REPLACE

DEFINE CHANNEL(SDSF.TO.J90) CHLTYPE(CLUSSDR) +
DESCR('CLUSTER SENDER CHANNEL') +
TRPTYPE(TCP) CONNAME(ip_address) +
CLUSTER(SDSF.CLUSTER) REPLACE

```

-
2. Selected and defined two systems to be repositories using the following MQ definition:

```
ALTER QMGR REPOS(SDSF.CLUSTER)
```

3. Updated the parameters in ISFPARMS to define SDSF clustering. We define our ISFPARMS in statement format in the ISFPRMxx member of SYS1.PARMLIB. We updated the COMM statement, which describes the communication for this server group. Our *pre-clustering* COMM statement looked as follows:

```
COMM NAME(COMM8),TYPE(MQS),QMGR(CSQ8),QPREFIX(ISF),QREPLACE(YES)
```

Our new clustered COMM statement looks as follows:

```
COMM NAME(COMM8),TYPE(MQS),QMGR(CQS8),QPREFIX(ISF),QREPLACE(YES),  
CLUSTER(SDSF.CLUSTER)
```

Note that we have specified a queue manager in the QMGR parameter. You can either do that, or you can specify a blank (with quotes around it) and define a default queue manager for your systems. We specify the queue manager name right on our COMM statement because we did not want to define a default queue manager for our systems. However, specifying the queue manager name on the COMM statement meant we had to delete all of our remote queue manager aliases. If you want to use queue manager aliases for other applications, you must do the following to allow SDSF to use the cluster rather than the queue manager aliases.

- Specify a blank (with quotes around it) in the QMGR parameter on the COMM statement, QMGR(' ').
- Run the CSQBDEFV program supplied by MQSeries to define a default queue manager. For complete instructions on specifying a default queue manager, see *Task 18: Set up Batch, TSO, and RRS adapters* in the *MQSeries for OS/390 System Setup Guide* book.

-
4. Recycled SDSF
-

Chapter 4. Implementing the DFSMSHsm common recall queue

We implemented a DFSMSHsm common recall queue (CRQ) environment in our sysplex. The CRQ environment allows multiple DFSMSHsm hosts to access a single recall queue to balance the recall workload across the HSMplex. The common recall queue uses a coupling facility list structure and any DFSMSHsm host that has a connection to the structure can perform the recalls. See *z/OS DFSMSHsm Storage Administration Guide*, SC35-0421, for a detailed description of CRQ processing.

Steps for defining the DFSMSHsm common recall queue environment

z/OS DFSMSHsm Implementation and Customization Guide, SC35-0418, provides instructions to implement the CRQ function. We used those instructions to perform the following steps:

1. Update the CFRM policy to define the CRQ list structure.

The structure name must be of the form `SYSARC_basename_RCL`, where *basename* is the five-character base name that you specify in the `SETSYS COMMONQUEUE(RECALL(CONNECT(basename)))` command in the `ARCCMDxx` parmlib member (see step 3 below). In our case, our structure name is `SYSARC_DFHSM_RCL`.

The size of the structure depends on the level of recall activity that occurs in the sysplex. Again, *z/OS DFSMSHsm Implementation and Customization Guide* provides guidance in selecting an appropriate size for the structure.

Our complete structure definition looks like this:

```
STRUCTURE NAME(SYSARC_DFHSM_RCL)
  MINSIZE(3840)
  SIZE(5120)
  PREFLIST(CF2,CF1,CF3)
  FULLTHRESHOLD(85)
  ALLOWAUTOALT(YES)
```

-
2. Activate the CFRM policy containing the new structure definition.

-
3. Update the DFSMSHsm `ARCCMDxx` member of parmlib with the appropriate `SETSYS` command.

In order to use the common recall queue across our HSMplex, we added the following `SETSYS` command to the `ARCCMDxx` member that all of our systems use during the startup of DFSMSHsm:

```
SETSYS COMMONQUEUE(RECALL(CONNECT(DFHSM)))
```

This completes the setup of the CRQ environment.

Using the DFSMSHsm common recall queue

After the setup of the CRQ environment is complete, CRQ processing can begin.

DFSMS common recall queue

Starting CRQ processing

When the SETSYS COMMONQUEUE(RECALL(CONNECT(*basename*))) command exists in the active ARCCMDxx member, DFSMShsm connects to the CRQ structure during startup processing. If you issue the SETSYS command after startup, DFSMShsm connects to the structure and then moves any requests on the host's local recall queue to the CRQ.

We saw the following messages when HSM connected to the CRQ structure:

```
IXL014I IXLCONN REQUEST FOR STRUCTURE SYSARC_DFHSM_RCL
WAS SUCCESSFUL.  JOBNAME: DFHSM ASID: asid
CONNECTOR NAME: HOSTCONNECTIONT CFNAME: CF1
```

```
ARC1501I CONNECTION TO STRUCTURE SYSARC_DFHSM_RCL WAS
ARC1501I (CONT.) SUCCESSFUL, RC=00, REASON=00000000
```

Displaying and controlling CRQ processing

The following is a summary of the new command options that allow you to display and control CRQ processing (see *z/OS DFSMSHsm Storage Administration Reference*, SC35-0422, for a complete explanation of each command):

QUERY COMMONQUEUE(RECALL)

Displays percent full, outstanding recall requests, and recall requests that are currently being processed in the common recall queue including what system is performing the recall

HOLD COMMONQUEUE

Prevents all CRQ processing—both placement of recall requests on the CRQ and selection of recall requests from the CRQ. Additional options offer more granular control over just the placement or selection of recall requests (see the DFSMSHsm documentation).

RELEASE COMMONQUEUE

Releases all holds on CRQ processing. Additional options offer more granular control over just the placement or selection of recall requests (see the DFSMSHsm documentation).

SETSYS COMMONQUEUE(RECALL(CONNECT(*basename*)))

Identifies the base name of the CRQ structure (SYSARC_*basename*_RCL) and allows HSM to connect to the structure.

SETSYS COMMONQUEUE(RECALL(DISCONNECT))

Directs recall processing back to the local recall queue and disconnects HSM from the CRQ structure.

Experiences with CRQ processing

We found it very easy to implement the CRQ environment and did not encounter any problems. Along with our regular HSM activity, we created a workload to generate more recall activity to fully exercise the CRQ function. We also successfully tested a variety of recovery scenarios for the CRQ structure.

Chapter 5. Migrating to CICS TS Version 2 Release 2

In this section, we describe our experiences with migrating to CICS Transaction Server Version 2 Release 2 (CICS TS 2.2). CICS TS 2.2 contains the following:

- CICS
- CICSplex System manager (CPSM) 2.2
- CICS Application Migration Aid
- CICS REXX (Runtime and Development)

Applicable documentation: We used information from the following publications to migrate to CICS TS 2.2 and CPSM 2.2:

- *CICS Transaction Server for z/OS CICS Operations & Utilities Guide*, SC34-5991
- *CICS Transaction Server for z/OS Installation Guide*, GC34-5985
- *CICS Transaction Server for z/OS Migration Guide*, GC34-5984
- *CICSplex SM Messages and Codes*, GC34-6028

Overview of migrating to CICS TS 2.2

One of our goals with this migration was to follow a plan that a typical customer might follow. To ensure a smooth migration, we migrated all of our CICS TS 1.3 regions to CICS TS 2.2 over a three-month period. We migrated our regions, one system at a time, starting on our test CICSplex. After we tested the migration steps on our test CICSplex, we were ready to migrate our production CICSplex, one region at a time.

Figure 6 shows our CICS TS 2.2 and CPSM 2.2 configuration:

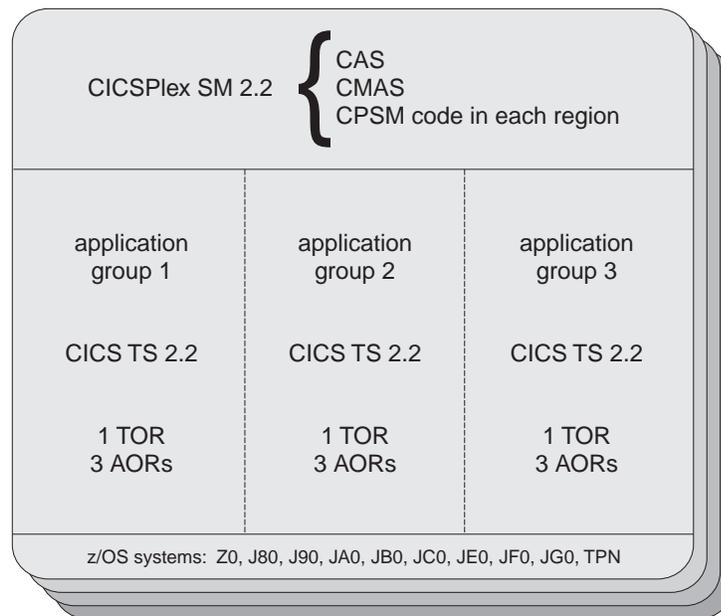


Figure 6. Our CICS TS 2.2 and CPSM 2.2 configuration

We run three distinct workloads under CPSM. So, we divided the migration into four stages: one for CPSM and one for each of the three workloads (CICS/VSAM-RLS, DB2, and IMS).

CICS TS

In the first stage, we migrated all of the CPSM components across the sysplex, one system at a time. On each system, CPSM consists of the following parts:

- CAS (coordinating address space)
- CMAS (CICS-managed address space)
- CPSM code in each CICS region, which communicates with the CMAS

Note: On any given z/OS system, all three parts of CPSM must be at the same release level. As long as all CPSM components on any given z/OS system are at the same level, you can run mixed levels of CPSM on different systems within a sysplex.

Upon completing the first stage, we had all of the CPSM components migrated to CICS TS 2.2, with all of the MASs (CICS regions) still running CICS TS 1.3.

After we were satisfied that the CPSM migration was stable and that there were no compatibility issues (between the CPSM releases moving across the sysplex and between the CICS releases on each system), we started on stage two—migrating the CICS/VSAM-RLS regions. Again, we migrated these regions on one system at a time. So, during the migration, we had a mixture of CICS releases communicating with CPSM.

We continued with stages three and four to migrate the DB2 and IMS workload regions in the same manner. We did not find any compatibility problems during the migration.

Performing the migration to CICS TS 2.2

This section describes in more detail how we migrated to CICS TS 2.2.

Preparing for migration

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

Backing up our data: As a precaution, we took backups of all of the CSDs and CICSplex SM data repositories.

Updating the LPA and LNKLST: We updated all of the LPA and LNKLST libraries for the new CICS TS 2.2 release level (CPSM 2.2 and CICS TS 2.2). All of the modules in these libraries need to be at the latest release of the code. We did this in advance of migration in order to ensure that the changes would not impact current systems. LNKLST and LPA modules at the CICS TS 1.3 level cannot run on a CICS TS 2.2 level system. If you try to migrate to CICS TS 2.2 without updating the LPA and LNKLST libraries, the following error message will result:

DFHIR3799 *applid* Unable to start interregion communication because DFHIRP services are down level.

APF-authorizing program libraries: We authorized the following CICS TS 2.2 libraries in SYS1.PARMLIB(PROGxx):

```
hlq.SDFHAUTH  
hlq.SDFHLINK  
hlq.SEYUAUTH  
hlq.SEYULINK
```

Allocating data sets: Next we decided to build as many of the region data sets as possible in advance. We did this primarily to address any DASD allocation issues. 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.

Migrating CICSplex SM

As with any of the CICS TS migrations, you must first migrate CPSM to the new 2.2 level before you migrate the rest of CICS TS. This is because the old level of CPSM does not recognize the new 2.2 release of CICS TS. To put this another way, CPSM 1.4.0 does not support CICS TS 2.2.

You must also migrate the maintenance point CMAS in CPSM first.

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. We defined a new BBIPARM parameter repository data set for CPSM 2.2.
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. We updated our TSO signon procedures to point to the new data sets for the new release of CPSM 2.2.

3. We reviewed the JCL in the EYUCAS member for any changes to the CAS startup procedure. Since 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. We started the CAS.

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

Note: With this release, you need to change the XCF group name. When the CAS first comes up, it takes a default group name of EYUGROUP. We changed our XCF group name to EYUGP220. In future releases, the generic default group name (EYUGROUP) will likely change to a release-specific name. As in any migration, CASs at different release levels cannot communicate with each other.

Migrating the CMASs

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

Steps for migrating the CMASs

We did the following to migrate the CMASs:

1. We defined a new CSD with a new record size.
In this release, there is a change in the CSD record size. You must define a new CSD with the correct record size (200 2000). The migration guide offers suggestions on how to accomplish this. We chose to define a new one with the proper record size and other attributes and then REPRO the old one into it.
2. We updated the CSD with CPSM 2.2 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 Transaction Server for z/OS CICS Operations & Utilities Guide*.
3. We updated the CICS system initialization table (SIT) GRPLIST parameter to point to the new CPSM 2.2 group list, EYU220G0.
4. We reassembled our CICS resource definition tables.
5. We converted the CPSM data repository to the CPSM 2.2 level by running the EYU9XDUT utility, as discussed in *CICS Transaction Server for z/OS Installation Guide*.
6. We reviewed the JCL in the EYUCMAS member for any changes to the CMAS startup procedure. Since 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. We updated the MAS JCL to point to the new CPSM data sets. This is to identify the new CMAS code to the MAS regions.

The CMAS and MAS regions on that system were now 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. We defined a new CSD with the new record size and REPROed the old CSD into the new one.

2. We upgraded the CSD using the group resource definitions in EYU220G1 for CPSM 2.2 and removed the group for the previous release of CPSM from the group list.

3. We reassembled our CICS resource definition tables.

Note: Although not required for the migration, we decided to switch from using a customized SIT to using the default SIT. This leaves us with one less table to reassemble. As a result, we needed to update our SYSIN SIT overrides accordingly.

4. We copied the JCL for our MAS startup procedure and changed the library names to use our new high-level qualifiers.

Our MASs were then ready to start.

Experiences with migrating to CICS TS 2.2

As we stated earlier, we did not encounter any compatibility issues during the entire migration process. However, we did run into a few operational problems:

Errors connecting a CMAS to a CAS using XCF communications: Last year, we changed our CAS-to-CAS communications from using VTAM to using XCF. We did this by changing two parameters in the CAS startup procedure: CONVXCF=Y and SPCF=Y. When we made that change under CICS TS 1.3, there were no problems, and we have been running that way ever since.

As soon as we tried to start a CICS TS 2.2 CAS and CMAS with those parameters, we received the following messages in the CMAS:

```
EYUXL0009I CVMP0C1 CAS connection established
BBMXC016E Error allocating TargetSvc RC: 8
Related:BBMXNS10E Too many properties to fit into 61K pipe state
Related:BBMXNS10E Too many properties to fit into 61K pipe state
Related:BBMXNS10E Too many properties to fit into 61K pipe state
Related:BBMXNS10E Too many properties to fit into 61K pipe state
```

When we tried to access a CPSM view from the TSO EUI, we received the following messages:

```
BBMXC716E Conversation allocation failed
BBMXBG11E The form Primary Query does not exist
```

The PTF for APAR PQ57280 resolves this problem.

CICS TS

Problem with CMASs shutting down unexpectedly: After we had CICS TS 2.2 up and running for a while, we found that a couple of our CMASs were shutting themselves down for no obvious reason (no abends, no dumps, and no MVS storage problems). After some investigation, we found the following error messages:

```
EYUXC0021S applid Total auxiliary storage limit has been reached  
EYUXC0024S applid The CMAS is terminating due to a previous error
```

This occurs when you use workload balancing along with abend compensation, real time analysis, or both. The PTF for APAR PQ57475 resolves this problem.

WLMAWORK view displays a duplicate workload: After a number of CMAS had been recycled, we happened to check the WLMAWORK view and we found what appeared to be a duplicate workload:

G1WLM	ACJ1	4	12	NONE	NONE	G1AOR	ACTIVE	N/A	QUE
G2WLM	ACJ1	1	17	NONE	NONE	G2AOR	ACTIVE	N/A	QUE
G2WLM	9CJ1	4	17	NONE	NONE	G2AOR	ACTIVE	N/A	QUE

Normally, during CMAS initialization, an attempt is made to connect the CMAS to any other active CMASs before connecting it to its MASs. The problem occurs when, because of a timing inconsistency, the CMAS connects to its MASs before it connects to other already active CMASs. The PTF for APAR PQ57915 resolves this problem.

Chapter 6. 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.

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 currently run 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 these products, see our December 2001 edition.

We are currently working on implementing msys for Operations, which actually includes parts of Tivoli NetView for OS/390 V1R4 and System Automation for OS/390 V2R1. We plan to document our experiences with msys for Operations in an upcoming edition of our test report.

Part 2. Networking and application enablement

Chapter 7. About our networking and application enablement environment	57
Our networking and application enablement configuration	57
Our Ethernet LAN configuration	58
Our ATM configuration	59
Our token ring LAN configuration	59
More about our backbone token ring	60
What's happening in LAN A?	60
What's happening in LAN B?	61
What's happening in LAN C?	62
Comparing the network file systems	64
Networking and application enablement workloads	65
Chapter 8. Using z/OS UNIX System Services	67
z/OS UNIX enhancements in z/OS V1R3	67
Using the automount enhancement	67
Using the enhanced configfs command	67
Using the BPXPRMxx UNMOUNT option for file system mounts	68
Using OMVS restart	68
Starting a z/OS UNIX colony address space outside of JES	69
Monitoring the z/OS UNIX mount table limit	71
Example of mount table limit monitoring in action	71
Example of switching OMVS couple data sets	73
Using access control lists (ACLs)	74
Overview of access control lists	74
About ACL terminology	75
Kinds of ACLs	75
Kinds of ACL entries	75
Some helpful tips	75
Working with access ACLs	76
Example of working with access ACLs	76
Working with default ACLs	79
Example of working with default ACLs	79
Auditing changes to ACLs	83
Example of auditing changes to ACLs	83
Using the getfacl and ls commands	86
Examples of using the getfacl and ls commands	86
Using the setfacl command	90
Example of using the setfacl command	91
A word about specifying options on the setfacl command	94
Using the find command with ACLs	94
Example of using the find command with ACLs	95
Using the pax and tar commands with ACLs	99
Example of using the pax and tar commands with ACLs	99
Using copytree and the cp and mv commands with ACLs	102
Example of using copytree and the cp and mv commands with ACLs	103
Using the df and getconf commands to determine file system ACL support	104
Examples of using the df command to determine file system ACL support	104
Examples of using the getconf command to determine file system ACL support	105
Using UNIXPRIV class profiles with ACLs	106
Using the RESTRICTED.FILESYS.ACCESS profile	106
Using the SUPERUSER.FILESYS.CHANGEPERMS profile	109
Using the SUPERUSER.FILESYS.ACLOVERRIDE profile	110

Using ISHELL with ACLs	112
Example of the Directory List and Display Attributes panels	112
Example of the Edit attributes menu.	113
Example of the Access Control List panel.	113
Managing a hierarchical file system (HFS)	115
Managing a zSeries file system (zFS)	115
zFS terminology	115
Setting up zFS	115
Chapter 9. Using IBM HTTP Server	121
Protecting IBM HTTP Server resources with LDAP	121
Customizing the HTTP server configuration file	121
Customizing the LDAP server configuration file	122
Verifying the LDAP protection for HTTP server resources	122
Steps to verify the LDAP protection for our protected resource	122
Chapter 10. Using LDAP Server	125
Overview of our LDAP configuration.	125
Enabling native authentication	126
Updating the schema for native authentication	126
Steps to update our schema	126
Learning from our mistakes	127
Updating the slapd.conf configuration file	127
Modifying directory entries to perform native authentication	128
Steps to modify one of our entries	128
Steps to verify our changes.	129
Enabling the IBM HTTP Server to exploit LDAP native authentication	130
Overview of HTTP server protection and LDAP native authentication	130
Steps to enable our IBM HTTP Server to exploit native authentication	130
Setting up a referral between LDAP databases	132
Setting up an LDAP server on Windows NT	132
Steps for setting up an LDAP server on Windows NT	132
Defining a referral in LDAP Server on z/OS	133
Steps for defining a referral in z/OS LDAP Server	133
Testing the referral function	133
Enabling LDAP for Kerberos authentication	134
Experiences with enabling LDAP for Kerberos authentication	134
Performing the setup steps	134
Setting up the environment for debugging	136
Verifying Kerberos authentication support in the LDAP server	136
Resolving a problem with malformed distinguished names in directory entries	137
Resolving the problem.	137
Chapter 11. Setting up Security Server Network Authentication Service	139
Experiences setting up Network Authentication Service.	139
Missing HFS directories	139
Creating the missing subdirectories under /etc	139
Creating the missing subdirectories under /var	140
Failure to consult the program directory for initial customization	140
Conflict with DCE	141
Steps to verify our Network Authentication Service customization	141
Chapter 12. Using Enhanced ASCII functionality	143
Overview of Enhanced ASCII	143
Understanding file tagging and automatic conversion	143
Implementing Enhanced ASCII	145

Working with Enhanced ASCII	146
Displaying file tag information	146
Controlling Enhanced ASCII functionality	147
Setting the AUTOCVT parameter in BPXPRMxx	147
C/C++ compiler and run-time support	148
Using the Language Environment FILETAG() run-time option	148
Using the TSO MOUNT command with the TAG() parameter	150
Using the mount shell command with the -c option	150
Using the mount panel under TSO ISHELL	151
Setting up automount to tag files	151
Using environment variables	152
Using redirection in /bin/sh and /bin/tcsh shell variables	153
Using the chtag command	154
Chapter 13. Using PKI Services	157
Introduction to PKI Services.	157
Installing and configuring prerequisite software.	157
Installing and configuring IBM HTTP Server	157
Installing and configuring OCSF and OCEP	158
Configuring OCSF	158
Configuring OCEP	159
Installing and configuring LDAP Server	159
Configuring the system for PKI Services	160
Running IKYSETUP to perform RACF administration	160
Steps for running IKYSETUP	161
Configuring the UNIX runtime environment	162
Tailoring the LDAP configuration for PKI Services.	163
Steps for tailoring the LDAP configuration for PKI Services	163
Updating the IBM HTTP Server configuration and starting the server	164
Updating the IBM HTTP Server configuration files	164
Starting the HTTP server instances	165
Tailoring the PKI Services configuration file for LDAP	166
Creating VSAM data sets and starting PKI Services	166
Updating and running IKYCVSAM to create VSAM data sets	166
Starting the PKI Services daemon	167
Customizing PKI Services	169
Customizing the end-user Web pages	169
Steps for performing minimal customization	169
Using PKI Services	170
Using the end-user Web pages	170
Steps for installing the CA certificate into the browser	170
Steps for creating a one-year SAF server certificate	170
Steps for creating a one-year SAF browser certificate	172
Steps for creating a two-year PKI browser certificate for authenticating to z/OS	173
Steps for creating a five-year PKI SSL server certificate	174
Chapter 14. Using IBM WebSphere Application Server for z/OS and OS/390	177
Packaging and deploying Web applications using WAR files	177
Steps for creating and deploying WAR files	177
Experiences with WAR files	178
Migrating Web applications to WebSphere Application Server V4.0 for z/OS and OS/390.	180
Migrating Web applications to the WebSphere for z/OS V4.0 plugin	180

Enabling WebSphere Application Server V4.0 for z/OS and OS/390 on a sysplex	181
Planning for WebSphere for z/OS and sysplex	182
Preparing the security system	182
Setting up data sharing	182
Customizing base z/OS functions on other systems in the sysplex	182
Updating the TCP/IP configuration	183
Setting up LDAP files for other systems in the sysplex	183
Defining the second WebSphere for z/OS system.	184
Steps to define our second WebSphere for z/OS system	184
Running the installation verification programs (IVP)	186
Updating the configuration for servlets that access MOFW or EJB applications from WebSphere for z/OS V4.0 plugin	187
Updating the startup procedure for the HTTP server.	187
Updating the httpd.conf file	188
Updating the httpd.envvars file.	188
Updating the was.conf file	188
Accessing our servlet	188
Migrating to WebSphere Application Server V4.0.1 for z/OS and OS/390	188
Steps to migrate from WebSphere for z/OS V4.0 to V4.0.1	189
Additional migration experiences with WebSphere for z/OS V4.0.1	192
Where to find more information	192
Chapter 15. Running Linux on IBM mainframe servers	193
Setting up Linux guests under z/VM.	193
Preparing the DASD	194
Creating a VM user definition for the first Linux guest	194
Setting up IP network connectivity	195
Building the first Linux instance	199
Creating and tailoring additional Linux instances	200
Setting up HiperSockets communications for LVMs	200
Defining the iQDIO hardware configuration	201
Adding HiperSockets links to the Linux virtual machine configuration.	202
Configuring the network interface under Linux	203
Steps to configure the network interfaces.	203
Experiences with HiperSockets communications	205
Setting up LPAR CPU management for non-z/OS partitions	205
Steps for setting up LPAR CPU management for our z/VM partition	205

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

Chapter 7. 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 7 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.

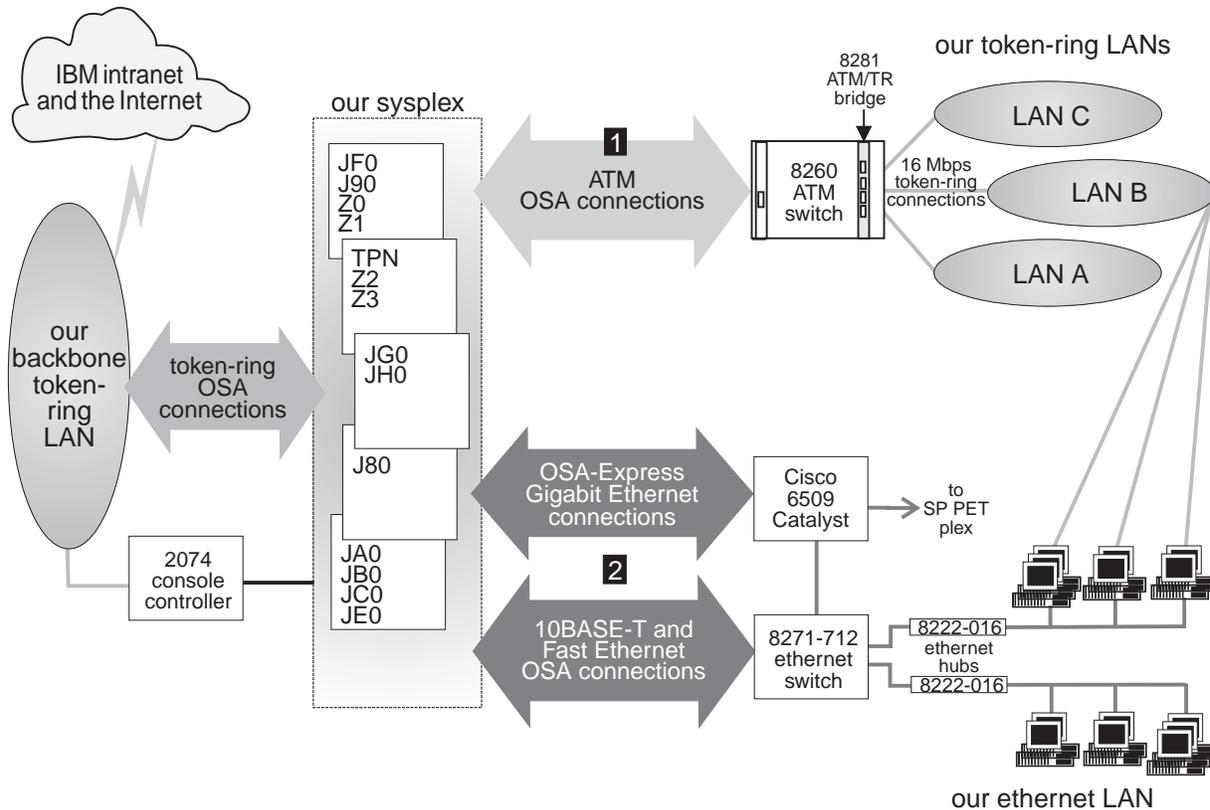


Figure 7. Our networking and application enablement configuration

Note the following about Figure 7:

- 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 14.

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 61

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 *S/390: 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 token ring LAN configuration

As Figure 7 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 7 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 7).

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 7; see Figure 10 on page 63 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.

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

Networking and applications environment

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 8 on page 61 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 7 above).

Networking and applications environment

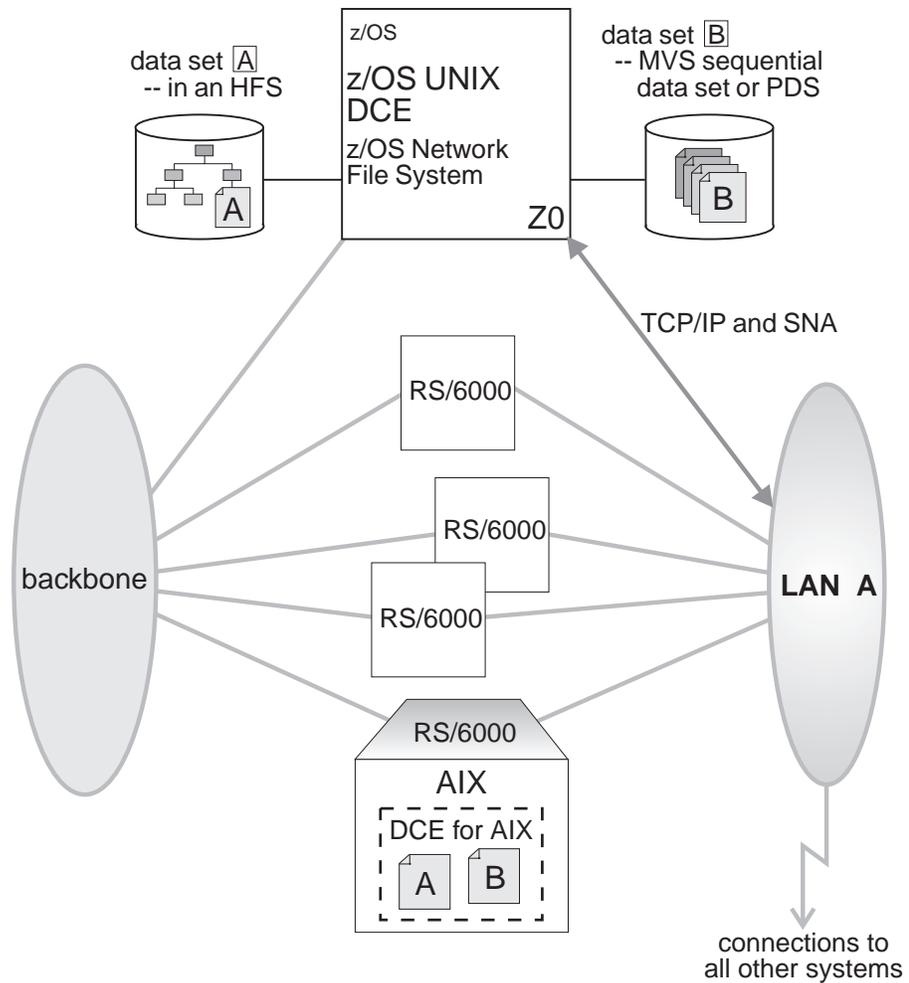


Figure 8. Our token-ring LAN A

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 15 for a more detailed description of these workloads. For more information about DCE and DFS, see “DCE and DFS Publications” in Appendix D, “Useful Web sites” on page 215.

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 64 for a description of the different types of NFSs we use.)

Figure 9 depicts our NFS and FTP test configuration in LAN B. For more information about NFS, see “Network File System Publications” in Appendix D, “Useful Web sites” on page 215.

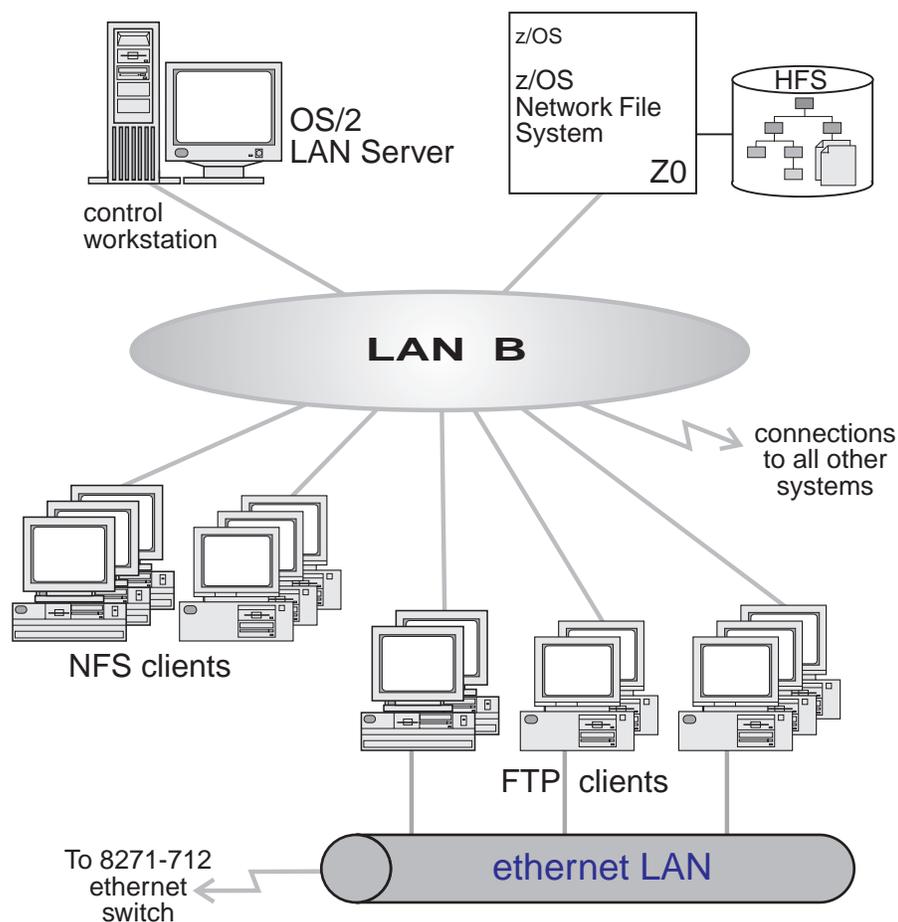


Figure 9. Our token-ring LAN B

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

Networking and applications environment

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 10 depicts our LAN Server test configuration in LAN C. For more information about LAN Server, see “LAN Server Publications” in Appendix D, “Useful Web sites” on page 215.

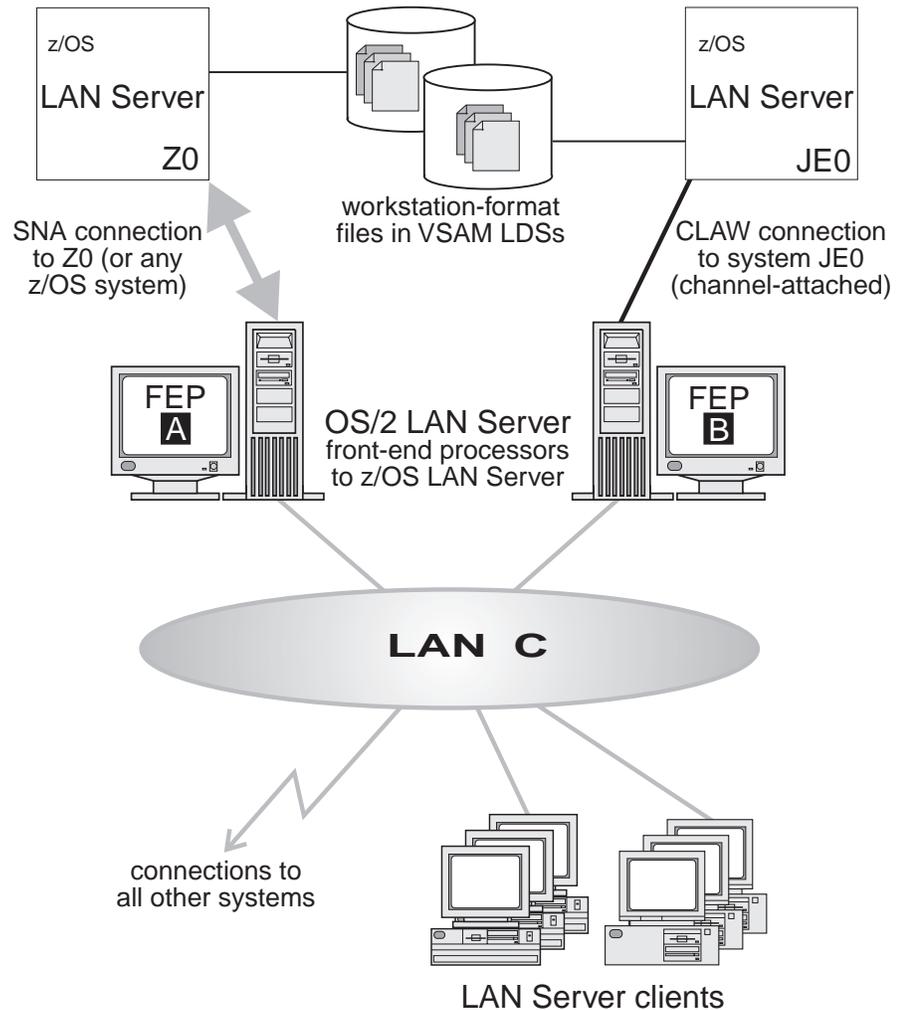


Figure 10. Our token-ring LAN C

Networking and applications environment

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, and the LAN Server NFS on system JE0. These two different NFSs cannot be connected to the same TCP/IP address space at the same time because they use the same port. So, a good way to avoid that is to run them on different systems. When we last reported on this configuration, we indicated that we ran them on the same system, using the same TCP/IP stack, but used them at different times. If you run your NFSs on different systems, you won't have to do that. The reason we changed to using them on different systems is that we implemented NFS server and client support for managing shared user home directory HFSs. See the HFS information in our December 1998 edition for more information.

Both of the NFSs 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.

With the LAN Server NFS, the data sets to be transferred to the workstations can reside only in special VSAM linear data sets that MVS users cannot read.

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. 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,

Networking and applications environment

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 15.

Chapter 8. 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 V1R3”
- “Using access control lists (ACLs)” on page 74
- “Managing a hierarchical file system (HFS)” on page 115
- “Managing a zSeries file system (zFS)” on page 115

z/OS UNIX enhancements in z/OS V1R3

In z/OS V1R3, z/OS UNIX made several enhancements:

- “Using the automount enhancement”
- “Using the enhanced confighfs command”
- “Using the BPXPRMxx UNMOUNT option for file system mounts” on page 68
- “Using OMVS restart” on page 68

We used *z/OS UNIX System Services Planning* for information about these enhancements.

Using the automount enhancement

For z/OS V1R3, z/OS UNIX can automatically allocate a new HFS in an automount managed directory, if it does not exist already, when a user enters the OMVS shell. The directory or directories you want to be automount managed are specified in the `auto.master` file – our automount managed directory is `/u`. You can use this function by specifying one of the two new keywords, **allocuser** and **allocany** in the `MapName` file. We tested both keywords in our `MapName` file, `/etc/u.map`:

- The **allocuser** keyword will cause z/OS UNIX to allocate an HFS if the name looked up matches the name of the current user. This is the keyword option we’re using in our permanent `u.map` file.
- The **allocany** keyword will cause z/OS UNIX to allocate an HFS if the data set specified does not exist for any name in the automount managed directory. Be careful when choosing to use the `allocany` keyword—it’s very easy to allocate unneeded HFSs by making typographical errors. Using `allocany`, when a user accessing directories managed by automount misspells the directory name, z/OS UNIX will allocate a new file system. We tested this successfully, but chose `allocuser` for our permanent `u.map` file.

For this to work in a shared HFS environment, the `/etc/u.map` file must be the same on each system in the sysplex.

Our permanent `/etc/u.map` file looks as follows:

```
name *
type HFS
filesystem OMVSSPN.USER.<uc_name>.FS
lowercase no
mode rdwr
allocuser space(1,1) cyl storclas(SMSOE)
duration 30
delay 00
```

Using the enhanced confighfs command

In z/OS V1R3, support for the `confighfs` command has been extended to client systems within a given shared HFS sysplex. Before V1R3, the `confighfs` command had to be issued from the system owning the HFS. Now however, the `confighfs` command can be issued from any client system in the shared HFS environment.

Using the BPXPRMxx UNMOUNT option for file system mounts

In z/OS V1R3, we tested and used the new UNMOUNT option for our system unique file system MOUNT statements. This means that when the system owning these file systems leaves the sysplex (for example, if the system fails or if the operator issues a soft shutdown) the file systems are unmounted rather than converted to unknown status. IBM recommends using the UNMOUNT option for system unique file systems.

The following example shows a MOUNT statement for one of our system unique file systems:

```
MOUNT FILESYSTEM('OMVSSPN.&SYSNAME..ETC.FS') TYPE(HFS)
      MODE(RDWR) MOUNTPOINT('/&SYSNAME./etc') UNMOUNT
```

Output from an MVS console DISPLAY command (D OMVS,F) issued against a file system mounted with the UNMOUNT parameter will look as follows:

```
HFS          445 ACTIVE
  NAME=OMVSSPN.JE0.ETC.FS
  PATH=/JE0/etc
  OWNER=JE0      AUTOMOVE=U CLIENT=N
```

Output from a \$ df -v command issued from the OMVS shell issued against a file system mounted with the UNMOUNT parameter will look as follows:

```
$ df -v /JE0/etc
Mounted on   Filesystem                Avail/Total   Files      Status
/JE0/etc    (OMVSSPN.JE0.ETC.FS)        76584/86400  4294966765 Available
HFS, Read/Write, Device:617, ACLS=Y
File System Owner : JE0      Automove=U    Client=Y
Filetag : T=off  codeset=0
```

Using OMVS restart

The OMVS restart function, which is new for z/OS V1R3, lets you restart z/OS UNIX without re-IPLing the z/OS system. To use this function, you first issue the F OMVS,SHUTDOWN command and then, when shutdown is complete, issue the F OMVS,RESTART command.

We tested the restart function as follows:

1. First, we stopped our z/OS UNIX workloads on the system where we planned to restart z/OS UNIX. We also shut down our automation programs, TCP/IP and zFS on the system. We did this to avoid the delays that can occur if blocking jobs or processes are active when a shutdown request is initiated. (A shutdown can be delayed until all blocking jobs or processes are either unblocked or ended.)

2. We issued the F OMVS,SHUTDOWN command. We received the following message in response:

```
*BPXI055I OMVS SHUTDOWN REQUEST ACCEPTED
GFSC701I OS/390 NFS CLIENT SHUTDOWN BEGIN
```

During some tests of this function, the system issued abend code SEC6 because there were threads active at the time of the shutdown.

3. Next we issued the D OMVS,A=ALL display command to check on the status of the shutdown. We received the following:

```
BPX0042I 14.22.07 DISPLAY OMVS 141
OMVS      000F SHUTTING DOWN 36  OMVS=(00,Z2)
```

As the shutdown progressed, file systems owned by the system were either moved to another system (AUTOMOVE) or unmounted (UNMOUNT), as set in the MOUNT parameter. This automove and unmount process is the same as when we issue the F BPXOINIT,SHUTDOWN=FILESYS command to take a system down for IPL.

-
4. After a short time, z/OS UNIX issues the following message to show that the shutdown has completed:

```
IEF196I BPXI056E OMVS SHUTDOWN REQUEST HAS COMPLETED SUCCESSFULLY
*BPXI056E OMVS SHUTDOWN REQUEST HAS COMPLETED SUCCESSFULLY
```

A display command (D OMVS,A=ALL) issued now also shows that z/OS UNIX has shut down:

```
D OMVS,A=ALL
BPX0042I 14.23.35 DISPLAY OMVS 192
OMVS      000F SHUTDOWN
```

-
5. Now we restart z/OS UNIX with the F OMVS,RESTART command. We receive the following messages:

```
IEF196I IEF237I BADF ALLOCATED TO IEFPARM
IEF196I IEF237I BADF ALLOCATED TO SYS00014
IEE252I MEMBER BPXPRMZ2 FOUND IN SYS1.PARMLIB
IEE252I MEMBER BPXPRM00 FOUND IN SYS1.PETR13.PARMLIB
IEF196I IEF285I   SYS1.PETR13.PARMLIB           KEPT
IEF196I IEF285I   VOL SER NOS= D83PET.
IEF196I IEF285I   SYS1.PARMLIB                 KEPT
IEF196I IEF285I   VOL SER NOS= D83PET.
IEF196I BPXI058I OMVS RESTART REQUEST ACCEPTED
```

zFS and NFS started up as specified in the settings in our BPXPRMxx members. All the file system mounts specified for that system in our BPXPRMxx members were processed, and our system unique file systems were mounted. Within a short time, z/OS UNIX completed the restart and issued the following message:

```
BPXI004I OMVS INITIALIZATION COMPLETE
```

-
6. After z/OS UNIX initialized, we restarted TCP/IP and our workloads. Our zFS and NFS subsystems are started automatically at startup as specified in our BPXPRMxx parameters.
-

Starting a z/OS UNIX colony address space outside of JES

Physical file systems can be set up to run in a colony address space. The name of the startup procedure must match the name specified in the ASNAME parameter of the FILESYSTYPE statement in the BPXPRMxx member that starts the physical file systems. By default, the colony address spaces are started under a JES subsystem. JES cannot be recycled without first bringing down all address spaces that were started under it, which disrupts use of the physical file system clients that

z/OS UNIX

use these address spaces. However, by entering an additional start parameter, SUB=MSTR, you can now start a colony address space outside of JES.

The ASNAME parameter of the FILESYSTYPE statement now has the following format:

```
ASNAME(procname, 'start_parms')
```

By entering SUB=MSTR in the *start_parms* field, you can have the physical file system colony start outside of JES.

The NFS Client, TFS, and zFS physical file systems support running outside of JES. However, the DFS Client does not currently support being started outside of JES control and, thus, does not support the SUB=MSTR start parameter.

Modifying the NFS Client and zFS startup procedures: When running outside of JES, you must make the following changes to the startup procedures for the NFS Client and zFS. Failure to make these changes will result in an S013 abend.

- Any DD statements that specify SYSOUT= need to be changed because SYSOUT data sets are only supported under JES. There are three alternatives for changing these data sets:
 - Direct the output to an MVS data set by changing to DD DSN=.
 - Direct the output to a z/OS UNIX file by changing to DD PATH=.
 - Discard the output by changing to DD DUMMY.
- There are some DD names that Language Environment (LE) opens under certain conditions. If these DD names are not allocated in the startup procedure, LE dynamically allocates them with SYSOUT=. (Exactly which DD names are opened, and when, varies by name, product, and the particular situation.) The DD names are:
 - SYSIN (for standard input)
 - SYSPRINT (for standard output -- if SYSPRINT does not exist, LE will look for and use SYSTERM or SYSERR if they exist, but LE will not dynamically allocate SYSTERM or SYSERR)
 - SYSOUT (for standard error, and also the default message file DD)
 - CEEDUMP (for capturing LE formatted dumps)

If any of these DD names are not currently used in the startup procedure for the colony address space, you must add them with DD DUMMY. Otherwise, any attempt to dynamically allocate them will result in an S013 abend.

Additional considerations when running outside of JES: The following are some additional consequences to consider when running a colony address space outside of JES:

- The colony address space will not be listed on SDSF displays.
- There will be no JOBLOG or system messages data set. System messages will go to SYSLOG.
- SMF recording is different between JES and the master subsystem.

For more information, see documentation APAR OW54086.

Also, if you intend to run colony address spaces outside of JES in a sysplex environment with mixed releases—that is, with z/OS V1R3 and earlier releases that support this function—ensure that you have installed the PTF for APAR OW48709.

Experiences with running colony address spaces outside of JES: We did the following to enable our NFS Client and zFS physical file system address spaces outside of JES:

- We modified our BPXPRMxx member of SYS1.PARMLIB as follows:

```

:
/* For zFS PFS, added SUB=MSTR for V1R3                */
FILESYSTYPE TYPE(ZFS) ENTRYPOINT(IOEFSCM) ASNAME(ZFS, 'SUB=MSTR')
:
/* For NFS Client, added SUB=MSTR for V1R3            */
FILESYSTYPE TYPE(NFS) ENTRYPOINT(GFSCINIT)
PARM('INITD,biod(6)') ASNAME(NFSOECLI, 'SUB=MSTR')
:

```

Note that we do not start a TFS as a colony address space.

- We changed our startup procedures for zFS and NFS Client to resolve the DD SYSOUT= concerns described earlier and in APAR OW54086. The changes were also necessary, so that we could use these procedures in a mixed sysplex environment (different z/OS releases), as long as the PTF for APAR OW48709 was installed on each release. Since, the procedures differ in each environment, refer to APAR OW54086 for the changes that are required for your installation.

Monitoring the z/OS UNIX mount table limit

A new BPXI043E eventual action console message will be issued to warn the operator when the number of mount table entries approaches the maximum limit defined in the OMVS couple data set (CDS). The message is issued when the number of mount table entries reaches 85 percent of the maximum. The message is again issued when the percentage reaches 90, 95 and 100. As the percentage increases to the next threshold level, the prior message is deleted from the console.

When the percentage falls below 85 and the resource shortage message has been deleted, a new message will be issued indicating that the resource shortage has been relieved. This is an informational message only and will not remain on the screen.

These messages are only issued when monitoring of parameter limits is active via the LIMMSG(SYSTEM) or LIMMSG(ALL) statement in the BPXPRMxx member of parmlib.

There are new fields in the BPXF041I message (in response to the MODIFY BPXOINIT,FILESYS=DISPLAY,GLOBAL command) in z/OS V1R3 that display the maximum and current number of mounts.

When a CDS swap occurs, the BPXI045I message will indicate the support of the mount limit and the limit of automount rules.

In our environment, we currently have the maximum mount entries set to 500. So, for example, when we issue the following command:

```
F BPXOINIT,FILESYS=DISPLAY,GLOBAL
```

the response message includes the following information:

```

:
MAXIMUM MOUNT ENTRIES=      500   MOUNT ENTRIES IN USE=      nnn
:

```

Example of mount table limit monitoring in action

In our environment, we set LIMMSG(SYSTEM) in our BPXPRMxx parmlib member. The following example demonstrates the monitoring behavior at each threshold.

z/OS UNIX

1. With 425 mounts active (85% of 500), the following highlighted message appeared:
BPXI043E MOUNT TABLE LIMIT HAS REACHED 85% OF ITS CURRENT CAPACITY OF 500

2. As we increased to 450 mounts active (90% of 500), the 85% message was unhighlighted and the following highlighted message appeared:
BPXI043E MOUNT TABLE LIMIT HAS REACHED 90% OF ITS CURRENT CAPACITY OF 500

3. With 475 mounts active (95% of 500), the 90% message was unhighlighted and the following highlighted message appeared:
BPXI043E MOUNT TABLE LIMIT HAS REACHED 95% OF ITS CURRENT CAPACITY OF 500

4. With 500 mounts active (100%), the 95% message was unhighlighted and the following highlighted message appeared:
BPXI043E MOUNT TABLE LIMIT HAS REACHED 100% OF ITS CURRENT CAPACITY OF 500

We verified this with the F BPX0INIT,FILESYS=DISPLAY,GLOBAL command.

Result:
:
:
MAXIMUM MOUNT ENTRIES= 500 MOUNT ENTRIES IN USE= 500
:
:

5. As we decreased to 475 mounts active (95%), the 100% message was unhighlighted and the following highlighted message appeared:
BPXI043E MOUNT TABLE LIMIT HAS REACHED 95% OF ITS CURRENT CAPACITY OF 500

6. With 450 mounts active (90%), the 95% message was unhighlighted and the following highlighted message appeared:
BPXI043E MOUNT TABLE LIMIT HAS REACHED 90% OF ITS CURRENT CAPACITY OF 500

7. With 425 mounts active (85%), the 90% message was unhighlighted and the following highlighted message appeared:
BPXI043E MOUNT TABLE LIMIT HAS REACHED 85% OF ITS CURRENT CAPACITY OF 500

8. As we fell below 425 mounts active, the following message appeared:
BPXI044I RESOURCE SHORTAGE FOR MOUNT TABLE HAS BEEN RELIEVED.

This concludes the example of mount table limit monitoring in action.

Example of switching OMVS couple data sets

The following example demonstrates how we formatted and switched to a new primary OMVS couple data set with different values for the maximum number of mounts and automount rules.

1. We formatted a new alternate OMVS couple data set to have the values of 501 for MOUNTS and 51 for AMTRULES.

```

:
ITEM NAME(MOUNTS) NUMBER(501)
:
ITEM NAME(AMTRULES) NUMBER(51)
:

```

2. We issued the command to make the alternate CDS our new primary CDS:

```
SETXCF COUPLE,PSWITCH
```

Result: The response includes the BPXI045I message to indicate the defined limits for the maximum number of mounts and automount rules.

```
BPXI045I THE PRIMARY CDS SUPPORTS A LIMIT OF 501 MOUNTS AND A LIMIT OF 51 AUTOMOUNT RULES
```

Note: All systems participate in the CDS swap. The first system to receive notification of the swap puts out the BPXI045I message (which might not be the same system from which the PSWITCH was issued).

3. We then formatted another alternate CDS and defined the values for MOUNTS and AMTRULES back to 500 and 50, respectively.

```

:
ITEM NAME(MOUNTS) NUMBER(500)
:
ITEM NAME(AMTRULES) NUMBER(50)
:

```

4. We issued the following command to try to activate the new alternate CDS:

```
SETXCF COUPLE,ACOUPLE=(SYS1.OMVS.CDS10,CDSOMP),TYPE=BPXMCD
```

Result: The command failed. You can activate a new CDS that has larger limits than those currently in effect, but you cannot go the other way; consistency checking prevents the activation of an alternate CDS whose item limits are smaller than the current primary CDS.

```
IXC309I SETXCF COUPLE,ACOUPLE REQUEST FOR BPXMCD WAS ACCEPTED
IXC260I ALTERNATE COUPLE DATA SET REQUEST FROM SYSTEM TPN FOR BPXMCD
IS NOW BEING PROCESSED.
```

```
DATA SET: SYS1.OMVS.CDS10
```

```
IEF196I IEF237I 2423 ALLOCATED TO SYS00109
```

```
IXC255I UNABLE TO USE DATA SET
```

```
SYS1.OMVS.CDS10
```

```
AS THE ALTERNATE FOR BPXMCD:
```

```
ALLOWABLE SIZE OF BPXFSMPT RECORDS IS LESS THAN CURRENT PRIMARY
```

```
IXC255I UNABLE TO USE DATA SET
```

```
SYS1.OMVS.CDS10
```

```
AS THE ALTERNATE FOR BPXMCD:
```

```
ALLOWABLE SIZE OF BPXFSAMT RECORDS IS LESS THAN CURRENT PRIMARY
```

```
IXC250I ALTERNATE COUPLE DATA SET REQUEST FAILED FOR DATA SET
SYS1.OMVS.CDS10 FOR BPXMCD5:
CONSISTENCY CHECKING FAILED FOR THE NEW ALTERNATE DATA SET
IEF196I IEF285I   SYS1.OMVS.CDS10                               KEPT
IEF196I IEF285I   VOL SER NOS= CDSOMP.
```

This concludes the example of switching OMVS couple data sets.

Using access control lists (ACLs)

In this section, we describe our experiences using access control lists (ACLs) to control access to files and directories.

Overview of access control lists

Access control lists are an extension to the standard POSIX permission bits which you can use to control access to files and directories by individual users (UIDs) and groups (GIDs). ACLs are created and checked by RACF (not the kernel or file system) and work in conjunction with the permission bits and other UNIXPRIV class profiles to determine file access. If you use a different security product, refer to its documentation to see if it supports ACLs and what rules it uses to determine file access.

Currently, the HFS and zFS file systems support ACLs. The file systems use the SAF `ck_access` (IRRSKA00) callable service to pass the ACL to the security product.

To manage ACLs, you must either be the file owner or have superuser authority (UID=0 or have READ access to SUPERUSER.FILESYS.CHANGEPERMS in the UNIXPRIV class). You can use the **setfacl** shell command to create, modify, and delete ACLs, and the **getfacl** shell command to display them. You can also use the ISHELL interface to define and display ACLs.

Applicable documentation: Refer to the following publications for more information about implementing and using ACLs:

- *z/OS UNIX System Services Planning*, GA22-7800
- *z/OS UNIX System Services Command Reference*, SA22-7802
- *z/OS Security Server RACF Security Administrator's Guide*, SA22-7683
- *z/OS Security Server RACF Auditor's Guide*, SA22-7684

In this section, we do not discuss the programming aspects of ACLs. For information about such topics, refer to the following publications:

- *z/OS Using REXX and z/OS UNIX System Services*, SA22-7806, documents several new syscall commands and new variables for some existing syscall commands to support ACLs.
- *z/OS C/C++ Run-Time Library Reference*, SA22-7821, documents the header files and new functions you can use to create, modify, and delete ACLs.
- *z/OS UNIX System Services Command Reference*, SA22-7802, documents new ACL primary operators on the **filetest**, **test**, **[...]**, and the **[[...]]** reserved-word commands.

About ACL terminology

access control list (ACL): (1) In computer security, a collection of all access rights for one object. (2) In computer security, a list associated with an object that identifies all the subjects that can access the object and their access rights; for example, a list associated with a file that identifies users who can access the file and identifies their access rights to that file. (3) In z/OS UNIX, an extension to the base POSIX permission bits. Similar to the access list of a RACF profile, an ACL for a file system object contains entries that specify access permissions for individual users and groups.

It is important to understand some basic terminology about ACLs and ACL entries. The following terms and definitions are key to understanding the information in the remainder of this section.

Kinds of ACLs

There are three kinds of ACLs:

- *Access ACLs* are ACLs that provide protection for a file system object.
- *File default ACLs* are default ACLs that are inherited by files created within the parent directory. A file inherits (from the directory in which it resides) the file default ACL as its access ACL. A subdirectory also inherits its parent directory's file default ACL as its file default ACL.
- *Directory default ACLs* are default ACLs that are inherited by subdirectories created within the parent directory. A subdirectory inherits its parent directory's default ACL as its directory default ACL and as its access ACL.

Notes:

1. Inheritance is the process of automatically associating an ACL with a newly created object. Administrative action is not needed.
2. The terms *default ACL* and *model ACL* are often used interchangeably.

Kinds of ACL entries

There are two kinds of ACL entries:

- *Base ACL entries* are the same as permission bits (owner, group, other). You can change the permissions using the **chmod** or **setfacl** commands. They are not physically part of the ACL although you can use **setfacl** to change them and **getfacl** to display them.
- *Extended ACL entries* are ACL entries for individual users or groups. Like the permission bits, they are stored with the file, not in RACF profiles. Each ACL type (access, file default, directory default) can contain up to 1024 extended ACL entries. Each extended ACL entry specifies a qualifier to indicate whether the entry pertains to a user or a group, the actual UID or GID itself, and the permissions being granted or denied by this entry. The allowable permissions are read, write, and execute. As with other UNIX commands, **setfacl** allows the use of either names or numbers when referring to users and groups.

Some helpful tips

The following are some helpful tips and recommendations about using ACLs:

- Analyze your HFS space utilization before implementing default ACLs in your file system. If you use both file and directory default ACLs in every directory in the file system, a separate physical ACL is created for every new file and directory.
- To minimize the performance impact, keep ACLs as small as possible, and grant permissions to groups instead of individual users. The pathlength of the access check increases with the size of an ACL, but will be shorter than the associated checking would be for a RACF profile with the same number of entries in its access list.

- Until you want ACLs to be used in access checks, make sure FSSEC is inactive. When you are ready to use ACLs, activate the FSSEC class by issuing the `SETROPTS CLASSACT(FSSEC)` command.
- ACLs can be defined and inherited while FSSEC is inactive. If you define default ACLs, they can be inherited by new objects when the FSSEC class is inactive. If the FSSEC class is inactive, the standard POSIX permission bit checking is done, even if an access ACL exists. You can also display ACL information while the FSSEC class is inactive.
- You may want to consider not disabling ACLs after you have been using ACLs for a while and have created many entries. If you have been using ACLs to grant, rather than deny, access to particular users and groups, then disabling ACLs may result in a loss of file access authority rather than a gain.
- ACLs are not inherited across mount points.

Example: Assume you have a default ACL defined on the `/dir1/dir2` directory. If you then create a new `/dir1/dir2/dir3` directory and use it as the mount point for another file system, then the root directory of the file system mounted at `/dir1/dir2/dir3` will *not* inherit the default ACL that was defined on `/dir1/dir2`. If you want the default ACLs of `/dir1/dir2` to apply to the root directory of the file system that is mounted at `/dir1/dir2/dir3`, then you must copy them to `/dir1/dir2/dir3` *after* the file system has been mounted there.

Note: ACLs are supported on AUTOMOUNTed directories (but are not inherited across mount points).

- An optional migration task for z/OS Security Server RACF is to upgrade any nodes that participate in shared HFS to z/OS V1R3 or, at a minimum, apply the compatibility APAR (OW50655 for SAF, and OW49334 for RACF) to the downlevel releases. Using ACLs should be no different in a sysplex as long as all of the participating systems are running z/OS V1R3 or higher. In a sysplex environment, all participating nodes must be on a release level that has ACL support. If any of the participating nodes are at a release level that does not contain ACL support and you have enabled the FSSEC class on an up-level node, then files that are protected by ACLs will not be accessible on down-level nodes (assuming that the compatibility APAR has been applied) except perhaps by a superuser or file owner.
- If you delete a file, ACLs associated with the file are automatically deleted.
- You need search access (execute permission) to display the ACL entries of a file using the `getfacl` command. However, if you only have search (execute) access (not read or write) for a directory (for instance, `drwx--x--x`), you cannot use the wildcard option for the files within that directory (such as `/u/alease1/file*`). Instead, you must fully qualify the path name, including the full file name (such as `/u/alease1/file_with_acl`).

Working with access ACLs

We'll begin with a simple example of using extended ACL entries for authorization checking on a single file.

Example of working with access ACLs

We logged in as a superuser (alease1) and did the following:

1. We used `oedit` to create a new file, entered some text into it, and then saved and filed it.

```
oedit acl_only_alease
```

2. We used the **ls** and **getfacl** commands to display the file's permissions. We have not created any extended ACL entries yet, so only the base ACL entries should appear.

Here's the **ls** command:

```
ls -alE acl_only_alease
```

Result: Notice that there is no plus sign (+) after the permissions because there are no extended ACL entries for this file.

```
-rwx----- --s- 1 alease1 sys1          16 mmm dd hh:ss acl_only_alease
```

Now the **getfacl** command:

```
getfacl acl_only_alease
```

Result: Notice that only the base ACL entries exist.

```
#file: acl_only_alease
#owner: alease1
#group: sys1
user::rwx
group:---
other:---
```

3. We used the **setfacl** command set (replaced) the access ACL for the file and defined extended ACL entries to do the following:

- Grant read and execute permission to user *alease*
- Deny access for group *employee*

```
setfacl -s user::rwx,group:---,other:---,user:alease:r-x,group:employee:--- acl_only_alease
```

4. We again displayed the file's permissions.

First the **ls** command:

```
ls -alE acl_only_alease
```

Result: Notice that there is now a plus sign (+) after the permissions to indicate that there are extended ACL entries.

```
-rwx-----+ --s- 1 alease1 sys1          16 mmm dd hh:mm acl_only_alease
```

Now the **getfacl** command:

```
getfacl acl_only_alease
```

Result: Notice that there are both base ACL entries and extended ACL entries.

```
#file: acl_only_alease
#owner: alease1
#group: sys1
user::rwx
group:---
other:---
user:ALEASE:r-x
group:EMPLOYEE:---
```

5. We activated the FSSEC class in RACF so that authorization checking will now include checking the extended ACL entries.

```
SETROPTS CLASSACT(FSSEC)
```

-
6. We turned off all of the base permissions so that access control uses only the extended entries in the file's access ACL. We used the **setfacl** command to do this (as shown below); however, you could also use the **chmod** command (for instance, `chmod 000 acl_only_alease`).

```
setfacl -s user::----,group::----,other::----,user:alease:r-x,group:employee:--- acl_only_alease
```

7. We displayed the file permissions again.

First the **ls** command:

```
ls -alE acl_only_alease
```

Result: Notice that the permissions are all set off.

```
-----+ --s- 1 alease1   sys1           16 mmm dd hh:mm acl_only_alease
```

Now the **getfacl** command:

```
getfacl acl_only_alease
```

Result: Notice that all permissions are now set off in the base ACL entries.

```
#file: acl_only_alease
#owner: alease1
#group: sys1
user::----
group::----
other::----
user:ALEASE:r-x
group:EMPLOYEE:---
```

We logged in with a non-superuser ID (alease9) and attempted to access the file.

8. We used the **obrowse** command to try to read the file.

```
obrowse /u/alease1/acl_only_alease
```

Result:

```
Errno=6Fx Permission is denied; Reason=5B4C0002
```

The error and reason codes mean the following:

```
111(006Fx) EACCES Permission is denied
5B4C GFUGOPEN, 0002 INVALID_OPEN
```

If the LOGOPTIONS for RACF class FSOBJ is set to at least DEFAULT (as with the command `SETROPTS LOGOPTIONS(DEFAULT(FSOBJ))`), the ICH408I message also appears in the system log, similar to the following:

```
ICH408I USER(ALEASE9 ) GROUP(SYS1   ) NAME(TESTER           )
/u/alease1/acl_only_alease
CL(FSOBJ   ) FID(01E2E2F0F0F4F1001E040000027D0000)
INSUFFICIENT AUTHORITY TO OPEN
ACCESS INTENT(R--) ACCESS ALLOWED(GROUP   ---)
EFFECTIVE UID(0000009000) EFFECTIVE GID(0000000000)
```

A similar ICH408I message would appear if we tried to access the file from a user ID in the EMPLOYEE group.

We logged in with another non-superuser ID (alease) and attempted to access the file. This is the user ID for which the file has an extended ACL entry that allows read (and execute) permission.

9. We used the **obrowse** and **oshell** commands to try to read the file.

First the **obrowse** command (from the z/OS UNIX shell):

```
obrowse /u/alease1/ac1_only_alease
```

Result: We were successfully able to browse the file.

Now the **oshell** command (from TSO or ISPF option 6):

```
oshell cat /u/alease1/ac1_only_alease
```

Result: We were successfully able to browse the file.

This concludes the example of working with access ACLs.

Working with default ACLs

As described earlier, default ACLs (or model ACLs) are automatically inherited by a file system object (a file or directory). Default ACLs have the same format as access ACLs. You can modify or delete an object's inherited ACLs later.

Example of working with default ACLs

In this example, we'll create a directory and define a file default ACL and a directory default ACL for it. Under this directory (and within the same file system) we'll then create a file and a subdirectory and observe the behavior of the interited ACLs. Finally, we'll demonstrate that ACLs are not inherited across mount points.

1. From within the /u/alease1 directory, we made a new directory and gave it a directory default ACL.

```
mkdir acldirD
setfacl -s d:user:alease:rwx,d:group:sys1:rwx acldirD
```

Result: We issued the command `getfacl -adf acldirD` to display the directory default ACL that we just defined.

```
#file: acldirD/
#owner: ALEASE1
#group: sys1
user::rwx
group::r-x
other::r-x
default:user:ALEASE:rwx
default:group:sys1:rwx
```

2. We also made a new subdirectory under the acldirD directory. The acldirD directory has a directory default ACL.

```
mkdir acldirD/acldirD2
```

Result: We issued the command `getfacl -adf acldirD/acldirD2` to display the ACLs for our new acldirD2 subdirectory. Notice that it inherited the directory default ACL from the acldirD directory as both its directory default ACL (**D**) and its access ACL (**A**).

```
#file: acldirD/acldirD2/
#owner: ALEASE1
#group: sys1
user::rwx
group::r-x
other::r-x
user:ALEASE:rwx
group:sys1:rwx
default:user:ALEASE:rwx
default:group:sys1:rwx
```

```
A
A
D
D
```

3. We deleted the directory default ACL from the acldirD directory and added a file default ACL, then displayed the results.

First we deleted the directory default ACL:

```
setfacl -vx d:user:alease:rwx,d:group:sys1:rwx acldirD
```

Then we added a file default ACL:

```
setfacl -vs f:user:alease:rwx,f:group:sys1:rwx acldirD
```

Result: We issued the command `getfacl -adf acldirD` to display the ACLs. Notice that the file default ACL (**F**) now exists.

```
#file: acldirD/
#owner: ALEASE1
#group: sys1
user::rwx
group::r-x
other::r-x
fdefault:user:ALEASE:rwx
fdefault:group:sys1:rwx
```

```
F
F
```

4. We used the oedit command to create a new file in the acldirD directory, saved the file, and then examined its ACLs.

```
oedit acldirD/acfileF
```

Result: We issued the command `getfacl -adf acldirD/acfileF` to display the ACLs. Notice that the file inherited the file default ACL from the acldirD directory as its access ACL (**A**).

```
FSUMF275 WARNING: file "acldirD/acfileF" is not a directory
#file: acldirD/acfileF
#owner: ALEASE1
#group: sys1
user::rwx
group:---
other:---
user:ALEASE:rwx
group:sys1:rwx
```

```
A
A
```

5. We again set a directory default ACL for the acldirD directory and modified the file default ACL, then displayed the results.

First we set the directory default ACL:

```
setfacl -vs d:user:alease:r--,d:group:sys1:r-- acldirD
```

Then we modified the file default ACL:

```
setfacl -vm f:user:alease:rwx,f:group:sys1:rwx acldirD
```

Result: We issued the command `getfacl -adf aclDirD` to display the ACLs. Notice that the directory now has both a file default ACL (**F**) and a directory default ACL (**D**).

```
#file: aclDirD/
#owner: ALEASE1
#group: sys1
user::rwx
group::r-x
other::r-x
fdefault:user:ALEASE:rwx
fdefault:group:sys1:rwx
default:user:ALEASE:r--
default:group:sys1:r--
```

```
F
F
D
D
```

6. We created a new file and made a new subdirectory under the `aclDirD` directory, then displayed the ACLs to observe how the default ACLs were inherited.

First we made a new subdirectory under the `aclDirD` directory:

```
mkdir aclDirD/aclDirD3
```

Next we created a new file in the `aclDirD` directory:

```
touch aclDirD/aclfile3
```

Result: We issued the command `getfacl -adf aclDirD/*` to display the ACLs for the objects under the `aclDirD` directory. Notice that the `aclDirD3` subdirectory inherited both its file default ACL (**F**) and directory default ACL (**D**) from the `aclDirD` directory; it also inherited the directory default ACL from `aclDirD` as its access ACL (**A**). Also, the `aclfile3` file inherited the file default ACL from `aclDirD` as its access ACL (**A**).

```
#file: aclDirD/aclDirD3/
#owner: ALEASE1
#group: sys1
user::rwx
group::r-x
other::r-x
user:ALEASE:r--
group:sys1:r--
fdefault:user:ALEASE:rwx
fdefault:group:sys1:rwx
default:user:ALEASE:r--
default:group:sys1:r--
```

```
A
A
F
F
D
D
```

```
FSUMF275 WARNING: file "aclDirD/aclfile3" is not a directory
```

```
#file: aclDirD/aclfile3
#owner: ALEASE1
#group: sys1
user::rw-
group::r--
other::r--
user:ALEASE:rwx
group:sys1:rwx
```

```
A
A
```

We continue our example by demonstrating that ACLs are not inherited across mount points. We use the same `aclDirD` and `aclDirD2` directories as above. We'll use the `aclDirD2` directory as the mount point for a new file system.

7. We used the **getfacl** command to display the current ACLs for the `acldirD` and `acldirD2` directories.

For `acldirD`, notice that a directory default ACL exists and contains entries for the user `alease` and the group `sys1`:

```
#file: acldirD/
#owner: ALEASE1
#group: sys1
user::r--
group::r--
other::r-x
fdefault:user:ALEASE:rwX
fdefault:group:sys1:rwX
default:user:ALEASE:rwX
default:group:sys1:rwX
```

For `acldirD2`, notice that this directory currently has ACL entries:

```
#file: acldirD/acldirD2/
#owner: ALEASE1
#group: sys1
user::rwX
group::r-x
other::r-x
user:ALEASE:rwX
group:sys1:rwX
default:user:ALEASE:rwX
default:group:sys1:rwX
```

8. We issued the following TSO MOUNT command to mount a new file system on the `/acldirD/acldirD2` mount point:

```
MOUNT FILESYSTEM('OMVSSPN.AGGR2.ZFS') TYPE(ZFS) MODE(RDWR)
      MOUNTPOINT('/u/alease1/acldirD/acldirD2')
```

9. We again displayed the permissions and ACLs for the `acldirD2` directory. Notice that only the base permissions exist. The mount on `acldirD/acldirD2` did not inherit the ACLs from `acldirD` or `acldirD/acldirD2`:

```
#file: acldirD/acldirD2/
#owner: ALEASE1
#group: sys1
user::rwX
group::--x
other::--x
```

10. If you want the newly-mounted file system on `acldirD/acldirD2` to have the same ACL definitions as `acldirD`, you can copy those ACLs to `acldirD2` after the file system has been mounted there. One way to do this is to issue the **getfacl** command on the `acldirD` directory and pipe the results to the **setfacl** command for the `acldirD2` directory, like this:

```
getfacl -adf acldirD | setfacl -vS - acldirD/acldirD2
```

Result: The command copies all three types of ACLs from one directory to the other.

```
Setting Access ACL entries for acldirD/acldirD2
Setting File Default ACL entries for acldirD/acldirD2
Setting Directory Default ACL entries for acldirD/acldirD2
```

-
11. We displayed the ACLs for `acldirD2` and saw that they were now the same as for `acldirD`:

```
#file: acldirD/acldirD2
#owner: ALEASE1
#group: sys1
user::r--
group::r--
other::r-x
fdefault:user:ALEASE:rwX
fdefault:group:sys1:rwX
default:user:ALEASE:rwX
default:group:sys1:rwX
```

This concludes the example of working with default ACLs.

Auditing changes to ACLs

Just as you can for the **chmod**, **chown**, and **chaudit** commands, you can use **SETROPTS LOGOPTIONS** for the **FSSEC** class to audit the creation, alteration, and deletion of ACLs. The **FSSEC** class controls auditing for changes to all file security information, including file owner, permission bits, and auditing options.

Two new event codes for SMF type 80 records support auditing ACLs:

Event code	Event
75 (4B)	SETFACL — creates one audit record for each ACL entry that is added, modified, or deleted
76 (4C)	DELFACL — creates one audit record

Note: You can find the RACF classes that control auditing for z/OS UNIX in *z/OS Security Server RACF Auditor's Guide*.

Example of auditing changes to ACLs

We did the following to test the auditing of changes to ACLs:

- From a user ID with the appropriate RACF authority, we made sure that the **FSSEC** class was active and then set the **LOGOPTIONS** for this class:

```
SETROPTS CLASSACT(FSSEC)
SETROPTS LOGOPTIONS(ALWAYS(FSSEC))
```

- From a z/OS UNIX superuser ID, we issued the following variety of commands to manipulate access control lists and cause audit records to be created:

- Create a new file:

```
touch /u/alease1/ac1smf
```

- Create an access ACL for the file:

```
setfacl -vs u::rwX,g::r-x,o::r-x,u:alease9:rwX,g:500:r-x /u/alease1/ac1smf
```

Result: Setting Access ACL entries for `/u/alease1/ac1smf`

- Modify the access ACL for the file:

```
setfacl -vm u:alease3:rwX,g:0:rwX /u/alease1/ac1smf
```

Result: Modifying Access ACL entries for `/u/alease1/ac1smf`

- d. Delete entries from the access ACL for the file:

```
setfacl -vx u:alease3:rxw,g:0:rxw /u/alease1/ac1smf
```

Result: Deleting Access ACL entries for /u/alease1/ac1smf

- e. Delete the access ACL for the file:

```
setfacl -vD a /u/alease1/ac1smf
```

Result: Deleting Access ACL for /u/alease1/ac1smf

- f. Attempt to delete the file default ACL:

```
setfacl -vD f /u/alease1/ac1smf
```

Result: FSUMF229 setfacl: warning: /u/alease1/ac1smf is not a directory so File Default ACL cannot be changed.

- g. Create a new directory:

```
mkdir /u/alease1/ac1smfdir
```

- h. Set the access ACL, file default ACL, and directory default ACL for the ac1smfdir directory:

```
setfacl -vs u::rxw,g::r-x,o::r-x,u:alease9:rxw,g:500:r-x,d:u:alease1:rxw,
d:g:0:rxw,f:u:0:rxw,f:g:0:rxw /u/alease1/ac1smfdir
```

Result:

```
Setting Access ACL entries for /u/alease1/ac1smfdir
Setting File Default ACL entries for /u/alease1/ac1smfdir
Setting Directory Default ACL entries for /u/alease1/ac1smfdir
```

- i. Delete the access ACL for the directory:

```
setfacl -vD a /u/alease1/ac1smfdir
```

Result: Deleting Access ACL for /u/alease1/ac1smfdir

- j. Delete the file default ACL for the directory:

```
setfacl -vD f /u/alease1/ac1smfdir
```

Result: Deleting File Default ACL for /u/alease1/ac1smfdir

- k. Delete the directory default ACL for the directory:

```
setfacl -vD d /u/alease1/ac1smfdir
```

Result: Deleting Directory Default ACL for /u/alease1/ac1smfdir

-
3. From a non-superuser ID, we issued the following variety of commands to attempt to manipulate the access control lists and cause audit records to be created:

- a. Set (replace) the access ACL for the ac1smf file:

```
setfacl -vs u::rxw,g::r-x,o::r-x,u:alease9:rxw,g:500:r-x /u/alease1/ac1smf
```

Result:

```
Setting Access ACL entries for /u/alease1/ac1smf
setfacl: FSUMF232 acl_set_file() failed on Access ACL for "/u/alease1/ac1smf":
return value:-1: EDC5111I Permission denied.
```

- b. Set (replace) the access ACL for the ac1smfdir directory:

```
setfacl -vs u::rxw,g::r-x,o::r-x,u:alease9:rxw,g:500:r-x /u/alease1/ac1smfdir
```

Result:

```
Setting Access ACL entries for /u/alease1/ac1smfdir
setfacl: FSUMF232 acl_set_file() failed on Access ACL for "/u/alease1/ac1smfdir":
return value:-1: EDC5111I Permission denied.
```

- c. Delete the access ACL for the file:

```
setfacl -vD a /u/alease1/ac1smf
```

Result:

```
Deleting Access ACL for /u/alease1/ac1smf
setfacl: FSUMF230 Unable to delete Access ACL from /u/alease1/ac1smf:
EDC5111I Permission denied.
```

- d. Delete the access ACL for the directory:

```
setfacl -vD a /u/alease1/ac1smfdir
```

Result:

```
Deleting Access ACL for /u/alease1/ac1smfdir
setfacl: FSUMF230 Unable to delete Access ACL from /u/alease1/ac1smfdir:
EDC5111I Permission denied.
```

-
4. We turned off logging and deactivated the FSSEC class:

```
SETROPTS LOGOPTIONS(NEVER(FSSEC))
SETROPTS NOCLASSACT(FSSEC)
```

-
5. We issued the SWITCH command to cause all in-storage SMF data to be written to the active SMF data set and transfer recording of SMF data to another data set.

```
I SMF
```

-
6. We submitted a job to run IRRADU00 to format and print the SMF records. The following is a portion of our JCL for this job:

```
//SMFDUMP EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=A
//ADUPRINT DD SYSOUT=A
//OUTDD DD DSN=&&TEMPSMF,UNIT=SYSDA,DISP=(NEW,PASS),
// SPACE=(CYL,(15,5)),
// DCB=(LRECL=5096,BLKSIZE=5100,RECFM=VB)
//SMFDATA DD DISP=SHR,DSN=SMFDATA.SMFTPN.G9419V00
//SMFOUT DD DUMMY
//SYSIN DD *
        INDD(SMFDATA,OPTIONS(DUMP))
        OUTDD(SMFOUT,TYPE(000:255))
        ABEND(NORETRY)
        USER2(IRRADU00)
        USER3(IRRADU86)
/*
//RMFSORT EXEC PGM=SORT,REGION=4M
//SORTIN DD DSN=&&TEMPSMF,DISP=(OLD,DELETE)
//SORTOUT DD SYSOUT=*
//SORTWK01 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(10))
//SORTWK02 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(10))
//SORTWK03 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(10))
//SYSPRINT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSIN DD *
```

z/OS UNIX

```
      SORT FIELDS=(11,4,CH,A,7,4,CH,A),EQUALS
      MODS E15=(ERBPPSRT,500),E35=(ERBPPSRT,500)
//SYSOUT DD   SYSOUT=H
//SYSPRINT DD  SYSOUT=H
```

The following are sample sections of the output from the job. In the second column, INSAUTH means that the event failed due to insufficient authority while SUCCESS means that the event successfully completed. Note that the output might contain records for other related commands, such as SETROPTS, CHOWN, and CHMOD.

```

:
:
DELFACL  INSAUTH  hh:mm:ss yyyy-mm-dd TPN  YES  NO  NO  ALEASE  SYS1.../u/alease1/ac1smf
DELFACL  INSAUTH  hh:mm:ss yyyy-mm-dd TPN  YES  NO  NO  ALEASE  SYS1.../u/alease1/ac1smfdir
:
:
SETFACL  INSAUTH  hh:mm:ss yyyy-mm-dd TPN  YES  NO  NO  ALEASE  SYS1.../u/alease1/ac1smf
SETFACL  INSAUTH  hh:mm:ss yyyy-mm-dd TPN  YES  NO  NO  ALEASE  SYS1.../u/alease1/ac1smfdir
:
:
DELFACL  SUCCESS  hh:mm:ss yyyy-mm-dd TPN  NO   NO  NO  ALEASE1  SYS1...ACCESS.../u/alease1/ac1smf
DELFACL  SUCCESS  hh:mm:ss yyyy-mm-dd TPN  NO   NO  NO  ALEASE1  SYS1...ACCESS.../u/alease1/ac1smfdir
DELFACL  SUCCESS  hh:mm:ss yyyy-mm-dd TPN  NO   NO  NO  ALEASE1  SYS1...FILEMOD.../u/alease1/ac1smfdir
DELFACL  SUCCESS  hh:mm:ss yyyy-mm-dd TPN  NO   NO  NO  ALEASE1  SYS1...DIRMOD.../u/alease1/ac1smfdir
:
:
SETFACL  SUCCESS  hh:mm:ss yyyy-mm-dd TPN  NO   NO  NO  ALEASE1  SYS1.../u/alease1/ac1smf
SETFACL  SUCCESS  hh:mm:ss yyyy-mm-dd TPN  NO   NO  NO  ALEASE1  SYS1.../u/alease1/ac1smfdir
:
:
```

This concludes the example of auditing changes to ACLs.

Using the getfacl and ls commands

In the previous examples, we used the **ls** command to display permissions (base ACL entries) and the **getfacl** command to display base and, if they exist, extended ACL entries for files and directories. We also saw that you can use the output of the **getfacl** command as input to the **setfacl** command. The **getfacl** command also resolves symbolic links.

Examples of using the getfacl and ls commands

In the following examples, we'll look at the behavior of the **getfacl** and **ls** commands under different conditions, such as the user's access permissions for the target file or directory.

Example 1—User has no permissions to the directory: In this example, a user (alease) has no permissions to the directory in which a file resides. The user is not allowed to display the file.

1. From the alease1 user ID, we removed permissions to the /u/alease1 directory for everyone except the owner (drwx-----):

```
chmod 700 /u/alease1
```

2. From the alease (non-superuser) user ID, we attempted to display the ACL information for the getfacl_alease_noaccess file in the /u/alease1 directory:

```
getfacl /u/alease1/getfacl_alease_noaccess
```

Result: The request failed due to insufficient authority. In the ICH408I message, notice that the request to display ACL information about a specific file

results in an access intent of EXECUTE (--X). Thus, the request minimally requires EXECUTE permission, but we currently have none.

```
FSUM6180 file "/u/alease1/getfac1_alease_noaccess": EDC5111I Permission denied.
```

```
ICH408I USER(ALEASE ) GROUP(SYS1 ) NAME(TESTER )
/u/alease1/getfac1_alease_noaccess
CL(DIRSRCH ) FID(01E2E2F0F0F4F1001E04000000000003)
INSUFFICIENT AUTHORITY TO LOOKUP
ACCESS INTENT(--X) ACCESS ALLOWED(GROUP ---)
EFFECTIVE UID(0013456712) EFFECTIVE GID(0000000000)
```

3. We attempted to display the base permissions and extended attributes for the same file:

```
ls -E /u/alease1/getfac1_alease_noaccess
```

Result: The command also failed due to insufficient authority.

```
ls: FSUM6785 File or directory "/u/alease1/getfac1_alease_noaccess" is not found
```

```
ICH408I USER(ALEASE ) GROUP(SYS1 ) NAME(TESTER )
/u/alease1/getfac1_alease_noaccess
CL(DIRSRCH ) FID(01E2E2F0F0F4F1001E04000000000003)
INSUFFICIENT AUTHORITY TO LOOKUP
ACCESS INTENT(--X) ACCESS ALLOWED(GROUP ---)
EFFECTIVE UID(0013456712) EFFECTIVE GID(0000000000)
```

Example 2—User has search (execute) permission to the directory: In this example, a user (alease) does have search (execute) permission to the directory in which a file resides. The user is not allowed to display the file by specifying a wildcard character in the file name; however, the user can display the file by specifying the full path and file name.

1. From the alease1 user ID, we gave search permission to the /u/alease1 directory to everyone (drwx--x--x):

```
chmod 711 /u/alease1
```

2. From the alease (non-superuser) user ID, we attempted to display the ACL information by specifying a wildcard character in the file name:

```
getfac1 /u/alease1/getfac1*
```

Result: The request failed due to insufficient authority. In the ICH408I message, notice that the request to display ACL information using a wildcard results in an access intent of READ (R--). Thus, the request minimally requires READ permission, but we only have EXECUTE permission.

```
FSUM6180 file "/u/alease1/getfac1*": EDC5129I No such file or directory.
```

```
ICH408I USER(ALEASE ) GROUP(SYS1 ) NAME(TESTER )
/u/alease1/. CL(DIRACC ) FID(01E2E2F0F0F4F1001E04000000000003)
INSUFFICIENT AUTHORITY TO OPENDIR
ACCESS INTENT(R--) ACCESS ALLOWED(GROUP --X)
EFFECTIVE UID(0013456712) EFFECTIVE GID(0000000000)
```

Note: We had LOGOPTIONS set to FAILURES for RACF classes DIRSRCH and DIRACC.

-
3. We tried again to display the ACL information using the full path and file name:
- ```
getfac1 /u/alease1/getfac1_alease_noaccess
```

**Result:** The request completed successfully. As we saw in the previous example, this request minimally requires EXECUTE permission, which we now have.

```
#file: /u/alease1/getfac1_alease_noaccess
#owner: setup
#group: sys1
user::rwx
group:---
other:---
user:ALEASE:r-x
```

---

**Example 3—Displaying extended ACL entries:** In this example, a user (alease1) owns a file with extended ACL entries, a directory with extended ACL entries (directory default ACL and file default ACL), and a symbolic link to the file that has the extended ACL entries.

1. We began by listing the directory, file, and symbolic link, along with their base permissions:

```
ls -alE acl*
```

**Result:**

```
drwxr-xr-x+ 2 ALEASE1 sys1 8192 Feb 13 10:31 acldir2
-rwxr-xr-x+ 1 ALEASE1 sys1 10 Feb 14 14:38 acl_a
lrwxrwxrwx 1 ALEASE1 sys1 9 Feb 19 14:15 acl_symlink -> acl_a
```

---

2. We issued the **getfac1** command to display ACL information for the acldir2 directory:

```
getfac1 -adf acldir2
```

**Result:** The ACL contains:

- Base ACL entries (user::, group::, other::)
- Extended ACL entries for users and groups (user:*name* and group:*name*)
- Extended file default ACL entries for users and groups (fdefault:user:*name* and fdefault:group:*name*)
- Extended directory default ACL entries for users and groups (default:user:*name* and default:group:*name*)

```
#file: acldir2/
#owner: setup
#group: sys1
user::rwx
group::r-x
other::r-x
user:setup:rwx
user:ALEASE:r-x
group:sys1:r-x
group:BIN:r-x
fdefault:user:setup:rwx
fdefault:user:ALEASE:r-x
fdefault:group:sys1:r-x
fdefault:group:BIN:r-x
```

```
default:user:setup:rwx
default:user:ALEASE:r-x
default:group:sys1:r-x
default:group:BIN:r-x
```

---

3. We displayed the ACL information for the `acl_a` file:

```
getfacl -adf acl_a
```

**Result:** The ACL contains base ACL entries and extended ACL entries (for two users and one group). The warning message occurs because we specified the `-d` and `-f` options but `acl_a` is not a directory. You can use the `-q` option (for instance, `getfacl -adfq`) to suppress the warning message.

```
FSUMF275 WARNING: file "acl_a" is not a directory
#file: acl_a
#owner: ALEASE1
#group: sys1
user::rwx
group::r-x
other::r-x
user:ALEASE9:r-x
user:ALEASE:---
group:sys1:r-x
```

---

4. We displayed the same ACL information as in step 3 but added the `-c` option:

```
getfacl -adfc acl_a
```

**Result:** The command response contains the same information except that the `-c` option uses commas to separate ACL entries, instead of listing each entry on a new line.

```
FSUMF275 WARNING: file "acl_a" is not a directory
user::rwx,group::r-x,other::r-x,user:ALEASE9:r-x,user:ALEASE:---,group:sys1:r-x
```

---

5. We displayed the ACL information for the `acl_symlink` symbolic link:

```
getfacl -adf acl_symlink
```

**Result:** The command resolved the symbolic link and displayed the contents of the ACL for the `acl_a` file. Notice that the base and extended entries are the same as those displayed earlier in step 3.

```
FSUMF275 WARNING: file "acl_symlink" is not a directory
#file: acl_symlink
#owner: ALEASE1
#group: sys1
user::rwx
group::r-x
other::r-x
user:ALEASE9:r-x
user:ALEASE:---
group:sys1:r-x
```

---

6. We repeated the command from step 5 and added the `-h` option:

```
getfacl -adfh acl_symlink
```

**Result:** No information was displayed. The -h option prevents the resolution of symbolic links. Since symbolic links themselves cannot have ACLs, the command returns no information.

---

**Example 4—Using `getfacl` with the `-e` option:** The `-e` option of the `getfacl` command limits the output to only those entries that pertain to the specified user (UID or user name). That is, the command only displays user entries for the specified user and group entries for the groups to which the user is connected. This can be useful in debugging access problems when files have large ACLs. You can limit the display even further by specifying `-a`, `-d`, or `-f` to target the type of ACLs (access, directory default, and file default, respectively) that you want to see.

In this example, we use the `-e` option to limit the `getfacl` display to only those ACL entries that pertain to user ALEASE.

1. We modified the access ACL for the `aclfile` file to add a user entry (`alease`) and two group entries (`sys1` and `employee`):

```
setfacl -m u:alease:r-x,g:sys1:r-x,g:employee:r-x aclfile
```

2. We displayed the ACL for the `aclfile` file. All ACL entries appear:

```
getfacl -adfqc aclfile
```

**Result:**

```
user::rwx,group::r-x,other::r-x,user:ALEASE:r-x,group:sys1:r-x,group:EMPLOYEE:r-x
```

3. We issued the `getfacl` command with the `-e` option to limit the display for user ALEASE:

```
getfacl -e alease *
```

**Result:** Besides the base permissions, only the extended ACL entries that pertain to user ALEASE appear—the entry for user ALEASE itself and the entry for the `sys1` group. User ALEASE is a member of the `sys1` group, so the entry for that group appears. User ALEASE is not a member of the `EMPLOYEE` group, so the entry for that group does not appear.

```
#file: aclfile
#owner: LORAIN0
#group: sys1
user::rwx
group::r-x
other::r-x
user:ALEASE:r-x
group:sys1:r-x
```

---

This concludes the examples of using the `getfacl` and `ls` commands.

## Using the `setfacl` command

The `setfacl` command sets (or replaces), modifies, or removes an ACL.

## Example of using the setfacl command

In this example, a user (alease1) sets base and extended ACL entries for a file and a directory, uses a symbolic link to a file to set extended ACL entries, and deletes ACL entries.

1. We listed the files and directories before we created any ACL entries:

```
ls -ald acl*
```

**Result:** Notice that there are no plus signs (+) after any of the base permissions, which indicates that there are no extended ACL entries.

```
drwxr-xr-x 2 ALEASE1 sys1 8192 Feb 13 10:31 acl_dir3
-rwxr-xr-x 1 ALEASE1 sys1 10 Feb 14 14:38 acl_a
-rwxr-xr-x 1 ALEASE1 sys1 10 Feb 14 14:38 acl_a_2
lrwxrwxrwx 1 ALEASE1 sys1 9 Feb 19 14:15 acl_symlink -> acl_a
```

2. We set (-s option) the required base ACL entries and one extended entry in the access ACL for the acl\_a\_2 file:

```
setfacl -vs u::rwx,g::r-x,o::r-x,u:alease9:rwx acl_a_2
```

**Result:** The verbose (-v) option displays the following response:

```
Setting Access ACL entries for acl_a_2
```

3. We modified (-m option) the ACL by adding extended entries for one user and one group to the access ACL:

```
setfacl -vm u:alease8:r-x,g:employee:r-- acl_a_2
```

**Result:**

```
Modifying Access ACL entries for acl_a_2
```

4. The following command uses the ACL entries from the acl\_a\_2 file to set the ACL for the acl\_a file (using its acl\_symlink symbolic link):

```
getfacl acl_a_2 | setfacl -vS - acl_symlink
```

**Result:** Symbolic links themselves cannot have ACLs. The acl\_symlink was resolved to the acl\_a file.

```
Setting Access ACL entries for acl_symlink
```

5. We displayed the ACL for the acl\_a\_2 file thus far:

```
getfacl -adfc acl_a_2
```

**Result:** The ACL entries reflect the entries we added and modified in steps 2 and 3 above.

```
FSUMF275 WARNING: file "setfacl_a" is not a directory
user::rwx,group::r-x,other::r-x,user:ALEASE8:r-x,user:ALEASE9:rwx,group:EMPLOYEE:r--
```

6. We displayed the ACL for the acl\_a file:

```
getfacl -adfcq acl_a
```

**Result:** The results match the ACL entries for the `acl_a_2` file (see step 5)—which we copied to the `acl_a` file in step 4. We also added the `-q` option to suppress the warning message.

```
user::rwx,group::r-x,other::r-x,user:ALEASE8:r-x,user:ALEASE9:rwx,group:EMPLOYEE:r--
```

---

7. We also displayed the ACL for the `acl_a` file using its symbolic link:

```
getfacl -adfcq acl_symlink
```

**Result:** The results match what we saw in step 6.

```
user::rwx,group::r-x,other::r-x,user:ALEASE8:r-x,user:ALEASE9:rwx,group:EMPLOYEE:r--
```

---

8. We set the file default ACL for user `alease9` on the `acldir3` directory:

```
setfacl -vs f:u:alease9:r-x acldir3
```

**Result:**

```
Setting File Default ACL entries for acldir3
```

---

9. We displayed the ACL for the `acldir3` directory:

```
getfacl -adfc acldir3
```

**Result:** The file default ACL entry appears.

```
user::rwx,group::r-x,other::r-x,fddefault:user:ALEASE9:r-x
```

---

10. We set the directory default ACL for user `alease9` on the `acldir3` directory:

```
setfacl -vs d:u:alease9:r-x acldir3
```

**Result:** The `-s` option sets (replaces) the contents of the ACL.

```
Setting Directory Default ACL entries for acldir3
```

---

11. We again issued the command from step 9 to display the ACL for the `acldir3` directory. Notice that the file default ACL entry (that we set in step 8) is gone. This is because, in step 10, we used the `-s` option (which replaces the ACL contents) instead of the `-m` option (which modifies the ACL contents).

```
user::rwx,group::r-x,other::r-x,default:user:ALEASE9:r-x
```

---

12. We then modified the base, access, file default, and directory default ACL entries for the `acldir3` directory:

```
setfacl -vm f:g:employee:r-x,f:u:alease9:r-x,d:g:bin:r-x,u::r-x,
 u:alease3:rwx,g:sys1:rwx acldir3
```

**Result:**

```
Modifying Access ACL entries for acldir3
Modifying File Default ACL entries for acldir3
Modifying Directory Default ACL entries for acldir3
```

13. We again issued the command from step 9 on page 92 to display the ACL for the `acldir3` directory. Notice that the `-m` option in step 12 added or replaced entries but did not remove any unaffected entries from the ACL.

```
user::r-x,group::r-x,other::r-x,user:ALEASE1:rwX,group:sys1:rwX,fdefault:user:ALEASE9:r-x,
fdefault:group:EMPLOYEE:r-x,default:user:ALEASE9:r-x,default:group:BIN:r-x
```

14. We deleted the access ACL entries from the ACL for the `acldir3` directory:

```
setfacl -vD a acldir3
```

**Result:**

Deleting Access ACL for `acldir3`

15. We again issued the command from step 9 on page 92 to display the ACL for the `acldir3` directory. Notice that the access ACL entries (`user:ALEASE1:rwX` and `group:sys1:rwX`) are now gone; however, the required base entries (`user::r-x,group::r-x,other::r-x`) remain.

```
user::r-x,group::r-x,other::r-x,fdefault:user:ALEASE9:r-x,fdefault:group:EMPLOYEE:r-x,
default:user:ALEASE9:r-x,default:group:BIN:r-x
```

16. We deleted the file default ACL entry for the `EMPLOYEE` group:

```
setfacl -vx fdefault:group:EMPLOYEE:r-x acldir3
```

**Result:**

Deleting File Default ACL entries for `acldir3`

17. We again displayed the ACL for the `acldir3` directory. Notice that only the `fdefault` entry for the `EMPLOYEE` group has been removed; the `fdefault` entry for `ALEASE9` remains intact.

```
user::r-x,group::r-x,other::r-x,fdefault:user:ALEASE9:r-x,default:user:ALEASE9:r-x,
default:group:BIN:r-x
```

18. We listed the directory and files again:

```
ls -ald acl*
```

**Result:** The directory and files (but not the symlink) have a plus sign following the base permissions to indicate that extended ACL entries exist.

```
dr-xr-xr-x+ 2 ALEASE1 sys1 8192 Feb 13 10:31 acldir3
-rwxr-xr-x+ 1 ALEASE1 sys1 10 Feb 14 14:38 acl_a
-rwxr-xr-x+ 1 ALEASE1 sys1 10 Feb 14 14:38 acl_a_2
lrwxrwxrwx 1 ALEASE1 sys1 9 Feb 19 14:15 acl_symlink -> acl_a
```

This concludes the example of using the `setfacl` command.

### A word about specifying options on the setfacl command

When you use the setfacl command to set, modify, or delete ACL entries, you must specify one of the required options (for instance, -s, -m, or -x) *after* any other options. Therefore, as an example, the following command is valid:

```
setfacl -vm u:alease9:rwx aclfile
```

However, the following command results in an error because the -m option does not follow all the other options:

```
setfacl -mv u:alease9:rwx aclfile
```

#### Result:

```
FSUMF209 setfacl: Cannot process the ACL data specified with the -m option.
Unable to parse text into an ACL: v
```

This requirement is illustrated in the syntax display for the **setfacl** command in *z/OS UNIX System Services Command Reference*, as follows:

```
setfacl [-ahqv] -s entries [path ...]
setfacl [-ahqv] -S file [path ... setfacl [-ahqv] -D type [...] [path ...]
setfacl [-ahqv] -m|M|x|X EntryOrFile [...] [path ...]
```

## Using the find command with ACLs

The example in this section demonstrates our testing of the **find** command with the following options that support ACLs:

| Option                        | Description                                                                                                                                                                                                                                                                                                    |
|-------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-acl c</b>                 | Matches if the type of ACL is the same as the type given by <i>c</i> , where <i>c</i> can be:<br><b>a</b> Access ACL (matches only if there are extended ACL entries)<br><b>d</b> Directory default ACL<br><b>f</b> File default ACL                                                                           |
| <b>-acl_count [+ -]number</b> | Matches if the numbers of extended ACL entries for any of the types of ACLs equals, is greater than (+), or is less than (-) <i>number</i> . The count applies to extended ACL entries <i>by type</i> (access, directory default, file default), <i>not</i> to the total number of extended ACL entries.       |
| <b>-acl_entry acl_text</b>    | Matches if the ACL contains an entry equivalent to <i>acl_text</i> , where <i>acl_text</i> is a single extended ACL entry.                                                                                                                                                                                     |
| <b>-acl_group groupid</b>     | Matches if the ACL contains an extended group ACL entry for <i>groupid</i> , where <i>groupid</i> can be a group name or a group ID number (GID). If a numeric group name (that is, a group name entirely made up of numeric digits) exists in the group database, the GID associated with that group is used. |
| <b>-acl_nogroup</b>           | Matches if a group ACL entry (for any type of ACL) exists but the actual group no longer exists on the system.                                                                                                                                                                                                 |
| <b>-acl_user userid</b>       | Matches if the ACL contains an extended user ACL entry for <i>userid</i> , where <i>userid</i> can be a user name or a user ID number (UID). If a                                                                                                                                                              |

numeric user name (that is, a user name entirely made up of numeric digits) exists in the user database, the UID associated with that user is used.

**-acl\_nouser** Matches if a user ACL entry (for any type of ACL) exists but the actual user no longer exists on the system.

When we first tested these options, we found some problems (which are now resolved) whereby files would be returned in error in certain cases. For example, the `-acl_nouser` and `-acl_nogroup` options would return files that had valid UIDs and GIDs, or requesting a certain GID would return files with a matching UID (and vice-versa). APAR OW53643 (PTF UW87075) resolves these problems.

### Example of using the find command with ACLs

This example demonstrates our testing of the `find` command with the ACL-related options.

To better understand the example, note that we have defined the following users and groups:

| User     | UID    |
|----------|--------|
| 00       | 666666 |
| 05       | 666666 |
| ALEASE9  | 9000   |
| OEDFLTU  | 666666 |
| setup    | 0      |
| Group    | GID    |
| EMPLOYEE | 500    |
| sys1     | 0      |

1. We created a new file, `aclfile`, and set its ACLs as follows:

```
setfacl -s user::rwx,group::r-x,other::r-x,user:0:rwx,group:0:r-x,user:9000:rwx,group:500:r-x aclfile
```

**Result:** Notice that when we display the ACLs for the `aclfile` file (below), the user names and group names appear even though we used the numeric UIDs and GIDs to create the ACLs.

```
#file: aclfile
#owner: setup
#group: sys1
user::rwx
group::r-x
other::r-x
user:setup:rwx
user:ALEASE9:rwx
group:sys1:r-x
group:EMPLOYEE:r-x
```

2. We issued several `find` commands and observed the results:

```
find . -acl_user 500
```

**Result:** No files returned. This is correct because our `aclfile` file has an entry for GID 500, but not for UID 500.

```
find . -acl_group 9000
```

**Result:** No files returned. This is correct because our `aclfile` file has an entry for UID 9000, but not for GID 9000.

```
find . -acl_user 9000
```

**Result:** There is an entry in the access ACL for aclfile for user ALEASE9 (UID 9000), so the file was returned:

```
./aclfile
```

```
find . -acl_group 500
```

**Result:** There is an entry in the access ACL for aclfile for group EMPLOYEE (GID 500), so the file was returned:

```
./aclfile
```

- 
3. We created a new file, `aclfilewithnumericusername`, and set its ACLs as follows:

```
setfacl -s u::rwx,g::rwx,o::rwx,u:05:rwx aclfilewithnumericusername
```

**Result:** When we display the ACLs, the numeric user name "05" (UID 666666) is resolved to OEDFLTU (UID 666666).

```
FSUMF275 WARNING: file "aclfilewithnumericusername" is not a directory
#file: aclfilewithnumericusername
#owner: ALEASE1
#group: sys1
user::rwx
group::rwx
other::rwx
user:OEDFLTU:rwx
```

- 
4. We modified the ACL for user name "00" as follows:

```
setfacl -vm u:00:rwx aclfilewithnumericusername
```

**Result:** Again, the system resolved the numeric user name "00" to its UID (666666). UID 666666 has the user name OEDFLTU and this is the name that appears in the ACL entry. Thus, rather than add a new ACL entry, this command modifies the same ACL entry that we created in step 3.

```
FSUMF275 WARNING: file "aclfilewithnumericusername" is not a directory
#file: aclfilewithnumericusername
#owner: ALEASE1
#group: sys1
user::rwx
group::rwx
other::rwx
user:OEDFLTU:rwx
```

- 
5. We issued the following two **find** commands:

```
find ~ -acl_user 05
find ~ -acl_user 00
```

**Result:** Both commands returned the same file. This is because both user name "00" and "05" have the same UID of 666666, and UID 666666 has the user name OEDFLTU, which matches an ACL entry for the `aclfilewithnumericusername` file.

```
/u/alease1/aclfilewithnumericusername
```

- 
6. We deleted the `aclfile` file and then re-created it and set its ACLs as follows:

```
setfacl -vs user::rwx,group::r-x,other::r-x,user:0:rwx,group:0:r-x,user:9000:rwx,group:500:r-x aclfile
```

**Result:** Again, when we display the ACLs for the file, the user names and group names appear instead of the numeric UIDs and GIDs that we used to define the ACLs:

```
#file: aclfile
#owner: setup
#group: sys1
user::rwx
group::r-x
other::r-x
user:setup:rwx
user:ALEASE9:rwx
group:sys1:r-x
group:EMPLOYEE:r-x
```

- 
7. We issued the following commands:

```
find . -acl_nogroup
find . -acl_nouser
```

**Result:** Both commands returned no files—there are no files whose ACLs contain entries for invalid users or groups.

- 
8. We deleted the `ALEASE9` user ID and the `EMPLOYEE` group from the system.

- 
9. We again issued the following commands:

```
find . -acl_nogroup
find . -acl_nouser
```

**Result:** Both commands returned the following file because its ACLs contain entries for a user and a group that no longer exist:

```
./aclfile
```

- 
10. We issued a variety of **find** commands, as follows:

- a. `find ~ -acl_entry user:0`

**Result:** Returns files that have an access ACL entry for user 0:

```
/u/alease1/aclfile
```

- b. `find ~ -acl_entry user:9000`

**Result:** Returns files that have an access ACL entry for user 9000:

```
/u/alease1/aclfile
```

- c. `find ~ -acl_entry f:user:9000`

**Result:** No files have a file default ACL entry for user 9000.

- d. `find ~ -acl_entry d:user:9000`

**Result:** No files have a directory default ACL entry for user 9000.

- e. `find ~ -acl_entry group:500`

**Result:** Returns files that have an access ACL entry for group 500:

```
/u/alease1/ac1file
```

- f. `find ~ -acl_entry f:group:500`

**Result:** No files have a file default ACL entry for group 500.

- g. `find ~ -acl_entry d:group:500`

**Result:** No files have a directory default ACL entry for group 500.

- h. `getfacl -adfcq ac1file`

**Result:** Displays the ACL for the ac1file file. Notice that the permissions for user ALEASE9 are set to r-x:

```
user::rwx,group::r-x,other::r-x,user:ALEASE9:r-x,user:ALEASE:---,group:sys1:r-x
```

- i. `find ~ -acl_entry user:alease9:+r^w`

**Result:** Returns files that have an access ACL entry for user ALEASE9 with read access and no write access:

```
/u/alease1/ac1file
```

- j. `setfacl -m user:alease9:rwx $(find ~ -acl_entry user:alease9:+r^w)`

**Result:** Modifies the access ACL entry for user ALEASE9 and sets the permissions to rwx for any files that currently have an access ACL entry for user ALEASE9 with read access and no write access.

- k. `getfacl -adfcq ac1file`

**Result:** Again displays the ACL for the ac1file file. The permissions for user ALEASE9 are now set to rwx:

```
user::rwx,group::r-x,other::r-x,user:ALEASE9:rwx,user:ALEASE:---,group:sys1:r-x
```

---

## 11. We created a directory, ac1countdir, and set its ACL entries as follows:

```
setfacl -vs u::rwx,g::r-x,o::r-x,u:alease1:rwx,u:alease:r-x,g:sys1:r-x,g:bin:r-x ac1countdir
setfacl -vm f:u:alease1:rwx,f:u:alease:r-x,f:g:sys1:r-x,f:g:bin:r-x ac1countdir
setfacl -vm d:u:alease1:rwx,d:u:alease:r-x,d:g:sys1:r-x,d:g:bin:r-x ac1countdir
```

**Result:** The following is the ACL display:

```
#file: ac1countdir/
#owner: ALEASE1
#group: sys1
user::rwx
group::r-x
other::r-x
user:ALEASE1:rwx
user:ALEASE:r-x
group:sys1:r-x
group:BIN:r-x
fdefault:user:ALEASE1:rwx
fdefault:user:ALEASE:r-x
fdefault:group:sys1:r-x
fdefault:group:BIN:r-x
default:user:ALEASE1:rwx
default:user:ALEASE:r-x
default:group:sys1:r-x
default:group:BIN:r-x
```

---

## 12. We issued a variety of **find** commands, as follows:

- a. `find ~ -acl_count 4 -print`

**Result:** Returns objects that have four entries in any type of ACL:

```
/u/alease1/ac1countdir
```

- b. `find ~ -acl a -acl_count 4 -print`  
**Result:** Returns objects that have four entries in the access ACL:  
/u/alease1/aclcountdir
- c. `find ~ -acl f -acl_count 4 -print`  
**Result:** Returns objects that have four entries in the file default ACL:  
/u/alease1/aclcountdir
- d. `find ~ -acl d -acl_count 4 -print`  
**Result:** Returns objects that have four entries in the directory default ACL:  
/u/alease1/aclcountdir
- e. `find ~ -acl_count +4`  
**Result:** No objects have more than four entries in any type of ACL.
- f. `find ~ -acl_count +3`  
**Result:** Returns objects that have more than three entries in any type of ACL:  
/u/alease1/aclcountdir
- g. `find ~ -acl_count -5`  
**Result:** Returns objects that have less than 5 entries in any type of ACL:  
/u/alease1/aclcountdir
- h. `find ~ -acl_count -5 -acl d`  
**Result:** Returns objects that have less than 5 entries in the directory default ACL:  
/u/alease1/aclcountdir

---

This concludes the example of using the **find** command with ACLs.

## Using the pax and tar commands with ACLs

For the **pax** command, we used the `-o saveext` option with the `-p A` or `-p e` option.

For the **tar** command, we used the `-U`, `-X`, and `-A` options.

**Note:** The documentation for the **tar** command may be missing the `-A` option to restore ACLs. Also, if **tar** is run with the `STORAGE` option on, the ACLs will not be restored and no error message is issued. APAR OW53639 resolves these problems.

### Example of using the pax and tar commands with ACLs

This example demonstrates our testing of the **pax** and **tar** commands to archive files with their ACLs and then restore the ACLs with the files. We show abbreviated command output to highlight the relevant points.

1. We used the **pax** command with the `-p A` option to archive the contents of the `/u/alease1` directory. We created an uncompressed archive and a compressed archive, and displayed the ACL information for both, as follows:
  - a. Created an uncompressed archive named `aclpaxA` containing the contents of `/u/alease1`:
 

```
cd /u/alease1
pax -o saveext -p A -wvf aclpaxA *
```
  - b. Created a compressed archive named `aclpaxz` containing the contents of `/u/alease1`:

```
pax -o saveext -p A -wzvf ac1paxz *
```

**Result:**

```
⋮
Compression: 57.96%
```

- c. Displayed the contents of both archives:

```
pax -vf ac1paxA
pax -vf ac1paxz
```

**Result:** The contents of both archives are the same:

```
⋮
drwxr-xr-x+ 1 WAJDA sys1 0 mmm dd hh:mm ac1dir2/
⋮
-rw-r--r--+ 1 WAJDA sys1 0 mmm dd hh:mm ac1shell2
```

- d. Displayed the ACL information for the contents of both archives:

```
pax -o A -vf ac1paxA
pax -o A -vf ac1paxz
```

**Result:** The ACL information is the same for both:

```
⋮
drwxr-xr-x+ 1 WAJDA sys1 0 mmm dd hh:mm ac1dir2/
user::rwx
group::r-x
other::r-x
fdefault:user:ALEASE9:r-x
⋮
-rw-r--r--+ 1 WAJDA sys1 0 mmm dd hh:mm ac1shell2
user::rw-
group::r--
other::r--
user:ALEASE9:rwx
user:ALEASE:rwx
group:sys1:r-x
group:EMPLOYEE:rwx
```

- 
2. We restored both the uncompressed archive and the compressed archive to a different file system and checked to see that the extended ACL entries were also restored, as follows:

- a. Prepared the new location for the restores:

```
cd /thrash2
mkdir ac1alease1
chmod 777 ac1alease1
cd ac1alease1
```

- b. Restored from the uncompressed archive using the -p A option to restore the ACLs:

```
pax -o saveext -p A -rvf /u/alease1/ac1paxA
```

**Result:** The **ls -al** command shows that extended ACL entries exist:

```
⋮
drwxr-xr-x+ 1 WAJDA sys1 0 mmm dd hh:mm ac1dir2/
⋮
-rw-r--r--+ 1 WAJDA sys1 0 mmm dd hh:mm ac1shell2
```

- c. Cleaned up the previous restore and restored from the compressed archive:

```
rm -rf /thrash2/aclalease1/*
pax -o saveext -p A -rvf /u/alease1/aclpaxz
```

**Result:** The **ls -al** command displayed the same results as for the previous restore.

- d. Cleaned up the previous restore and restored from the uncompressed archive using the **-p e** option:

```
rm -rf /thrash2/aclalease1/*
pax -o saveext -p e -rvf /u/alease1/aclpaxA
```

**Result:** The **ls -al** command displayed the same results as for the previous restore.

- e. Cleaned up the previous restore and restored from the compressed archive:

```
rm -rf /thrash2/aclalease1/*
pax -o saveext -p e -rvf /u/alease1/aclpaxz
```

**Result:** The **ls -al** command displayed the same results as for the previous restore.

3. We used the **pax** command with the **-p e** option to archive the contents of the **/u/alease1** directory. We created an uncompressed archive and a compressed archive, and displayed the ACL information for both, as follows:

- a. Created an uncompressed archive and a compressed archive:

```
pax -o saveext -p e -wvf aclpaxA /u/alease1/*
pax -o saveext -p e -wzvf aclpaxz /u/alease1/*
```

- b. Displayed the contents of both archives:

```
pax -vf aclpaxA
pax -vf aclpaxz
```

**Result:** The displays showed that the contents of both archives are the same and that extended ACL entries exist (+ signs are present).

- c. Displayed the ACL information for the contents of both archives:

```
pax -o A -vf aclpaxA
pax -o A -vf aclpaxz
```

**Result:** The same ACL information appears in both archives.

4. As we did earlier in step 2, we restored the uncompressed archive and the compressed archive (with the **-o saveext** and **-p e** options and again with the **-o saveext** and **-p A** options, cleaning up the restored files in between each attempt) and checked to see that the extended ACL entries were also restored. Each attempt successfully restored the ACL data.

5. We used the **tar** command with the **-UX** options to archive the contents of the **/u/alease1** directory. We displayed the contents of the archive and the ACL information in the archive, as follows:

- a. Created an uncompressed archive named **actar** containing the contents of **/u/alease1**:

```
cd /u/alease1
tar -cvUXzf actar *
```

**Result:**

Compression: 77.88%

- b. Displayed the contents of the archive:

```
tar -vtf acltarz
```

**Result:** The contents of the archive includes extended ACL entries:

```

:
drwxr-xr-x+ 1 WAJDA sys1 0 mmm dd hh:mm acldir2/
:
-rw-r--r--+ 1 WAJDA sys1 0 mmm dd hh:mm aclshell2

```

- c. Displayed the ACL information for the contents of the archive:

```
tar -L A -vtf acltarz
```

**Result:** As we saw with the **pax** command, the ACL information remains intact:

```

:
drwxr-xr-x+ 1 WAJDA sys1 0 mmm dd hh:mm acldir2/
user::rwx
group::r-x
other::r-x
fdefault:user:ALEASE9:r-x
:
-rw-r--r--+ 1 WAJDA sys1 0 mmm dd hh:mm aclshell2
user::rw-
group::r--
other::r--
user:ALEASE9:rwx
user:ALEASE:rwx
group:sys1:r-x
group:EMPLOYEE:rwx

```

- 
6. We restored from the archive using the **-A**, **-U**, and **-X** options to restore the ACLs:

```
cd /thrash2
tar -xpAUXvf /u/alease1/acltarz
```

**Result:** The **ls -al** command shows that extended ACL entries exist:

```

:
drwxr-xr-x+ 1 WAJDA sys1 0 mmm dd hh:mm acldir2/
:
-rw-r--r--+ 1 WAJDA sys1 0 mmm dd hh:mm aclshell2

```

---

This concludes the example of using the **pax** and **tar** commands with ACLs.

## Using copytree and the cp and mv commands with ACLs

Copytree is a REXX sample that shows how to use a number of useful z/OS UNIX capabilities. It includes a recursive routine to descend a hierarchical directory, retrieving and setting attributes for files, reading and writing files, and reading and setting access control lists (ACLs). z/OS UNIX now includes copytree in the /samples directory.

**Note:** In TSO, copytree cannot handle files that are greater than 1 GB in size. In the z/OS UNIX shell, it requires the REXX function package for files greater than 1 GB.

### Example of using copytree and the cp and mv commands with ACLs

This example demonstrates our testing of the copytree tool and the **cp** and **mv** commands to copy files along with their ACLs.

1. We issued the following command to copy files from the /u/alease1 file system to the /thrash2 file system:

```
/samples/copytree /u/alease1 /thrash2
```

**Result:**

```
Copying /u/alease1 to /thrash2
Scanning for file nodes...
Skipping mountpoint: /u/alease1/..
Processing 71 nodes
Creating directories
Creating other files
Setting file attributes
```

```

```

```
Copy complete. Error count= 0
Directory errors: 0
Directories copied: 8
File errors: 0
Files copied: 61
Symlink errors: 0
Symlinks copied: 2
Char-spec errors: 0
Char-spec copied: 0
FIFO errors: 0
FIFOs copied: 0
Sparse file count: 0
ACL count: 34
```

- 
2. We verified the results by issuing the **getfacl** command against the source and target file systems and comparing the results:

```
getfacl -adf /u/alease1/*
getfacl -adf /thrash2/*
```

**Result:** Both commands displayed the same ACL entries.

- 
3. We issued the following command to copy a file along with its ACL entries:

```
cp -p aclishell aclishellcp
```

**Note:** The -p option of the **cp** command preserves the ACLs of files and directories, if possible. The ACLs are not preserved if a file system does not support ACLs or if you copy files to MVS data sets.

**Result:** We issued the **getfacl** command against the source and the target and the results were the same.

- 
4. We issued the following command to move a file:

```
mv aclishell12 /zfsfs1/aclishell12cpfromhfs
```

**Result:** We issued the **getfactl** command against the source and the target and the results were the same.

---

This concludes the example of using copytree and the **cp** and **mv** commands with ACLs.

## Using the **df** and **getconf** commands to determine file system ACL support

The following examples demonstrate our testing of the **df** and **getconf** commands to determine whether different types of file systems (HFS, TFS, and zFS) support ACLs.

### Examples of using the **df** command to determine file system ACL support

The **-v** option of the **df** command includes information about whether the security product and file system support ACLs.

The following examples demonstrate the use of the **df** command to determine ACL support:

1. The following command displays information about the TFS:

```
df -vkP /tmp/tfs
```

**Result:** The TFS does not support ACLs (ACLS=N).

```
Filesystem 1024-blocks Used Available Capacity Mounted on
/Z0/TMP/TFS 10000 82 9918 1% /Z0/tmp/tfs
TFS, Read/Write, Device:547, ACLS=N
-s 10
File System Owner : Z0 Automove=U Client=N
Filetag : T=off codeset=0
```

2. The following command displays information about the HFS:

```
df -vkP ~
```

**Result:** The HFS supports ACLs (ACLS=Y).

```
Filesystem 1024-blocks Used Available Capacity Mounted on
OMVSSPN.USER.ALEASE1.FS 7920 6660 1136 86% /u/alease1
HFS, Read/Write, Device:645, ACLS=Y
File System Owner : TPN Automove=Y Client=N
Filetag : T=off codeset=0
```

3. The following command displays information about the zFS:

```
df -vkP /zfsfs
```

**Result:** The zFS supports ACLs (ACLS=Y).

```
Filesystem 1024-blocks Used Available Capacity Mounted on
OMVSSPN.AGGR2.ZFS 9575 22 9553 1% /zfsfs
ZFS, Read/Write, Device:575, ACLS=Y
File System Owner : TPN Automove=Y Client=N
Filetag : T=off codeset=0
```

---

This concludes the examples of using the **df** command to determine file system ACL support.

### Examples of using the **getconf** command to determine file system ACL support

You can use the **getconf** command to display the values of the `_PC_ACL` and `_PC_ACL_ENTRIES_MAX` configuration variables. The `_PC_ACL` variable indicates whether the security product supports ACLs. The `_PC_ACL_ENTRIES_MAX` variable indicates the maximum number of entries that you can place in an ACL for a specified file.

The following examples demonstrate the use of the **getconf** command to determine ACL support:

1. Display the `_PC_ACL` variable for the TFS:

```
getconf _PC_ACL /tmp/tfs
```

**Result:** 0 (The TFS does not support ACLs.)

---

2. Display the `_PC_ACL` variable for the HFS:

```
getconf _PC_ACL ~
```

**Result:** 1 (The HFS supports ACLs.)

---

3. Display the `_PC_ACL` variable for the zFS:

```
getconf _PC_ACL /zfsfs
```

**Result:** 1 (The zFS supports ACLs.)

---

4. Display the `_PC_ACL_ENTRIES_MAX` variable for the TFS:

```
getconf _PC_ACL_ENTRIES_MAX /tmp/tfs
```

**Result:** The TFS does not support ACLs:

```
getconf: Invalid argument (_PC_ACL_ENTRIES_MAX or /tmp/tfs)
```

---

5. Display the `_PC_ACL_ENTRIES_MAX` variable for the HFS:

```
getconf _PC_ACL_ENTRIES_MAX ~
```

**Result:** 1024 (The HFS supports a maximum of 1024 entries in an ACL.)

```
1024
```

---

6. Display the `_PC_ACL_ENTRIES_MAX` variable for the zFS:

```
getconf _PC_ACL_ENTRIES_MAX /zfsfs
```

**Result:** 1024 (The zFS supports a maximum of 1024 entries in an ACL.)

---

This concludes the examples of using the **getconf** command to determine file system ACL support.

## Using UNIXPRIV class profiles with ACLs

The following examples demonstrate how we tested file access behavior using ACLs in conjunction with the following UNIXPRIV class profiles:

- RESTRICTED.FILESYS.ACCESS
- SUPERUSER.FILESYS.CHANGEPERMS
- SUPERUSER.FILESYS.ACLOVERRIDE

### Using the RESTRICTED.FILESYS.ACCESS profile

Users with the RESTRICTED attribute in RACF cannot access protected resources that they are not specifically authorized to access. However, the RESTRICTED attribute has no effect when a user accesses a z/OS UNIX file system resource; the file's "other" permission bits can allow access to users who are not explicitly authorized. To ensure that restricted users do not gain access to z/OS UNIX file system resources through "other" bits, you can use the RESTRICTED.FILESYS.ACCESS profile of the UNIXPRIV class. This can be overridden for a specific user or group via READ access to the RESTRICTED.FILESYS.ACCESS profile.

**Example of using the RESTRICTED.FILESYS.ACCESS profile:** In this example, we create a RESTRICTED user ID, ALEASE8. After defining the RESTRICTED.FILESYS.ACCESS profile, we try to access a file (via BPXBATCH) that is neither owned by nor in the same group as user ID ALEASE8, but that does have its permission bits on for "other" users. This should not allow ALEASE8 to access the file. We then permit ALEASE8 to the RESTRICTED.FILESYS.ACCESS profile, which should allow access through the "other" permission bits. After deleting ALEASE8's permission to the RESTRICTED.FILESYS.ACCESS profile, we define an extended access ACL for user ALEASE8 to allow read, write, and execute permission. After activating the FSSEC RACF class, access should be granted through the ACL entries only.

We performed the following steps:

1. We issued the following commands from a user ID that is authorized to issue RACF commands:

```
RDEFINE UNIXPRIV RESTRICTED.FILESYS.ACCESS UACC(NONE)
SETROPTS RACLIST(UNIXPRIV) REFRESH
```

2. We issued the following commands from a user ID other than ALEASE8:

```
echo otherpermactext > otherpermacc
chown alease1:3 otherpermacc
chmod 007 otherpermacc
```

```
ls -alE otherpermacc
```

**Result:** The "other" permission bits for the otherpermacc file are set to allow read, write, and execute access.

```
-----rwx --s- 1 ALEASE1 3 65 mmm dd hh:mm otherpermacc
```

---

3. We submitted the following BPXBAL1 job which attempts to read the otherpermacc file:

```
//BPXBAL1 JOB ('BPXBATCH-BACKGROUND USS'),D10.PETHFSN.JOBS,
// MSGLEVEL=(1,1),PERFORM=10,MSGCLASS=I,CLASS=K,TIME=1440,
// USER=ALEASE8,PASSWORD=password
//*MAIN SYSTEM=JGLOBAL
/*JOBPARM SYSAFF=*
//BPXBATCH EXEC PGM=BPXBATCH,
// PARM='SH cat /fsfull/otherpermacc'
//STEPLIB DD DSN=ALEASE8.SYS1.LINKLIB,DISP=SHR
//STDOUT DD PATH='/u/alease8/myrstd.out.zfs',
// PATHOPTS=(OWRONLY,OCREAT,OTRUNC),PATHMODE=SIRWXU
//STDERR DD PATH='/u/alease8/myrstd.err.zfs',
// PATHOPTS=(OWRONLY,OCREAT,OTRUNC),PATHMODE=SIRWXU
//
```

**Result:** Read access to the otherpermacc file was not allowed because user ALEASE8 has the RESTRICTED attribute.

```
ICH408I USER(ALEASE8) GROUP(SYS1) NAME(TESTER)
 /fsfull/otherpermacc
CL(FSOBJ) FID(0000001100000015000186A000000005)
INSUFFICIENT AUTHORITY TO OPEN
ACCESS INTENT(R--) ACCESS ALLOWED(RESTRICTED ---)
EFFECTIVE UID(0000008123) EFFECTIVE GID(0000000000)
ICH408I USER(ALEASE8) GROUP(SYS1) NAME(TESTER)
IEDCMSGT CL(PROGRAM)
INSUFFICIENT ACCESS AUTHORITY
FROM * (G)
ACCESS INTENT(READ) ACCESS ALLOWED(NONE)
CSV025I PROGRAM CONTROLLED MODULE IEDCMSGT NOT ACCESSED, USER UNAUTHORIZED
```

4. We submitted the following BPXBAL2 job which attempts to write to the otherpermacc file:

```
//BPXBAL2 JOB ('BPXBATCH-BACKGROUND USS'),D10.PETHFSN.JOBS,
// MSGLEVEL=(1,1),PERFORM=10,MSGCLASS=I,CLASS=K,TIME=1440,
// USER=ALEASE8,PASSWORD=password
//*MAIN SYSTEM=JGLOBAL
/*JOBPARM SYSAFF=Z3
//BPXBATCH EXEC PGM=BPXBATCH,
// PARM='SH echo alease8write >> /fsfull/otherpermacc'
//STEPLIB DD DSN=ALEASE8.SYS1.LINKLIB,DISP=SHR
//STDOUT DD PATH='/u/alease8/mywstd.out.zfs',
// PATHOPTS=(OWRONLY,OCREAT,OTRUNC),PATHMODE=SIRWXU
//STDERR DD PATH='/u/alease8/mywstd.err.zfs',
// PATHOPTS=(OWRONLY,OCREAT,OTRUNC),PATHMODE=SIRWXU
//
```

**Result:** Write access to the otherpermacc file was not allowed because user ALEASE8 has the RESTRICTED attribute.

```
ICH408I USER(ALEASE8) GROUP(SYS1) NAME(TESTER)
 /fsfull/otherpermacc
CL(FSOBJ) FID(0000001100000015000186A000000005)
INSUFFICIENT AUTHORITY TO OPEN
ACCESS INTENT(-W-) ACCESS ALLOWED(RESTRICTED ---)
EFFECTIVE UID(0000008123) EFFECTIVE GID(0000000000)
ICH408I USER(ALEASE8) GROUP(SYS1) NAME(TESTER)
IEDCMSGT CL(PROGRAM)
```

```

INSUFFICIENT ACCESS AUTHORITY
FROM * (G)
ACCESS INTENT(READ) ACCESS ALLOWED(NONE)
CSV025I PROGRAM CONTROLLED MODULE IEDCMSGT NOT ACCESSED, USER UNAUTHORIZED

```

---

Using the RESTRICTED.FILESYS.ACCESS profile, you can grant exceptions to selected RESTRICTED users and allow them to gain access to file system resources based on the setting of the “other” permission bits. You do this by adding a user (or one of the groups in which it is a member) to the access list of the profile with READ authority.

5. We issued the following RACF commands to permit user ALEASE8 to the RACF profile RESTRICTED.FILESYS.ACCESS with READ access to now allow this user to access file system resources based upon their “other” permission bits:

```

PERMIT RESTRICTED.FILESYS.ACCESS CLASS(UNIXPRIV) ID(ALEASE8) ACCESS(READ)
SETROPTS RACLIST(UNIXPRIV) REFRESH

```

---

6. We again submitted the BPXBAL1 job (which attempts read access) and the BPXBAL2 job (which attempts write access) to the otherpermacc file.

**Result:** The jobs ran successfully and were able to read and write, respectively, to the otherpermacc file as allowed by the value of its “other” permission bits.

---

7. We issued the following RACF commands to delete user ALEASE8 from the access list of the RESTRICTED.FILESYS.ACCESS profile:

```

PERMIT RESTRICTED.FILESYS.ACCESS CLASS(UNIXPRIV) ID(ALEASE8) DELETE
SETROPTS RACLIST(UNIXPRIV) REFRESH

```

---

8. We created an entry for user ALEASE8 in the access ACL for the otherpermacc file to grant the user read, write, and execute access:

```
setfacl -vs u:---,g:---,o::rwx,u:alease8:rwx otherpermacc
```

**Result:** We used the **getfacl** command to display the ACL data that we set:

```

#file: otherpermacc
#owner: ALEASE1
#group: 3
user:---
group:---
other:rwx
user:ALEASE8:rwx

```

The results from the **ls** command look like this:

```
-----rwx+ --s- 1 ALEASE1 3 65 mmm dd hh:mm otherpermacc
```

---

9. We issued the following RACF command to activate the FSSEC class to enable ACL checking:

```
SETROPTS CLASSACT(FSSEC)
```

Note that user ALEASE8 still has the RESTRICTED attribute in RACF and is no longer permitted to the RESTRICTED.FILESYS.ACCESS profile. However, access checking based on ACLs is now active.

10. We again submitted the BPXBAL1 and BPXBAL2 jobs to attempt to read from and write to the otherpermacc file.

**Result:** The jobs ran successfully and were able to read from and write to the otherpermacc file as allowed by the entry in the file's access ACL.

This concludes the example of using the RESTRICTED.FILESYS.ACCESS profile and ACLs to selectively permit restricted users to access file system resources.

### Using the SUPERUSER.FILESYS.CHANGEPERMS profile

The SUPERUSER.FILESYS.CHANGEPERMS profile in the UNIXPRIV class allows users to issue the **chmod** command to change the permission bits of any file and to issue the **setfacl** command to manage the ACLs for any file. To manage ACLs for a file, a user must either be the file owner or have superuser authority (that is, have a UID of 0 or have READ access to the SUPERUSER.FILESYS.CHANGEPERMS profile).

**Example of using the SUPERUSER.FILESYS.CHANGEPERMS profile:** We've seen many examples so far of the use of the **setfacl** command by file owners and superusers. In this example, we demonstrate how we tested the ability to allow a user who is a non-owner and non-superuser to manage ACLs by permitting the user to the SUPERUSER.FILESYS.CHANGEPERMS profile.

We did the following:

1. From user ALEASE9, we tried to modify the ACL for the aclishell file in /u/alease1 (owned by user alease1). The alease9 user is not a superuser and is not the owner of the file.

```
setfacl -vm g:500:r-x /u/alease1/aclishell
```

**Result:** The attempt to modify the ACL was unsuccessful.

```
Modifying Access ACL entries for /u/alease1/aclishell
setfacl: FSUMF232 acl_set_file() failed on Access ACL for "/u/alease1/aclishell"
: return value:-1: EDC5111I Permission denied.
```

2. We issued the following RACF commands to permit the ALEASE9 user to the SUPERUSER.FILESYS.CHANGEPERMS profile in the UNIXPRIV class:

```
RDEFINE UNIXPRIV SUPERUSER.FILESYS.CHANGEPERMS UACC(NONE)
PERMIT SUPERUSER.FILESYS.CHANGEPERMS CLASS(UNIXPRIV) ACCESS(READ) ID(ALEASE9)
SETROPTS RACLIST(UNIXPRIV) REFRESH
```

3. From the ALEASE9 user ID, we again attempted to modify the ACL for the file owned by alease1:

```
setfacl -vm g:500:r-x /u/alease1/aclishell
```

**Result:** This time the command successfully completed. We used the **getfacl** command to display the results of our change (the EMPLOYEE group has GID 500):

```
#file: /u/alease1/aclishell
#owner: ALEASE1
#group: sys1
user::rw-
group::r--
other::r--
user:ALEASE9:rwX
user:ALEASE:rwX
group:sys1:r-x
group:EMPLOYEE:r-x
```

- 
4. Still using the ALEASE9 user ID, we tried modifying the base permission bits with the **chmod** command.
- We used the **ls** command to display the current permission bits:
 

```
-rwx-----+ 1 ALEASE1 sys1 10 mmm dd hh:mm /u/alease1/setfac1_a
```
  - We issued the **chmod** command to modify the permission bits:
 

```
chmod 755 /u/alease1/setfac1_a
```

**Result:** The command completed successfully. We displayed the permissions following the change:

```
-rwxr-xr-x+ 1 ALEASE1 sys1 10 mmm dd hh:mm /u/alease1/setfac1_a
```

We used the **getfacl** command to display the results:

```
#file: /u/alease1/setfac1_a
#owner: ALEASE1
#group: sys1
user::rwX
group::r-x
other::r-x
user:ALEASE9:r-x
user:ALEASE:---
group:sys1:r-x
```

---

This completes the example of using the SUPERUSER.FILESYS.CHANGEPERMS profile to permit non-owners and non-superusers to modify permissions and manage ACLs.

### Using the SUPERUSER.FILESYS.ACLOVERRIDE profile

A user (not a superuser or file owner) who is denied access to a file system resource through the ACL can still access the resource if the user has sufficient authority to the SUPERUSER.FILESYS profile in the UNIXPRIV class in RACF. To prevent this, you can use the SUPERUSER.FILESYS.ACLOVERRIDE profile to force RACF to use your ACL authorizations to override a user's SUPERUSER.FILESYS authority.

**Example of using the SUPERUSER.FILESYS.ACLOVERRIDE profile:** In this example, we demonstrate the use of the SUPERUSER.FILESYS.ACLOVERRIDE profile to enforce the ACL authorizations, overriding a user's SUPERUSER.FILESYS authority.

We did the following:

- We started by making sure that no ACL overrides were in effect and that user ALEASE (non-superuser, no RACF attributes) has no access to the SUPERUSER.FILESYS profile. We issued the following RACF commands:

```
RDELETE UNIXPRIV SUPERUSER.FILESYS.ACLOVERRIDE
SETROPTS RACLIST(UNIXPRIV) REFRESH
PERMIT SUPERUSER.FILESYS CLASS(UNIXPRIV) ACCESS(NONE) ID(ALEASE)
SETROPTS RACLIST(UNIXPRIV) REFRESH
```

---

2. From the ALEASE user ID, we tried to read the file /u/alease1/alease1file, which has the following permissions:

```
-rwxr-x--- 1 ALEASE1 EMPLOYEE 7 mmm dd hh:mm alease1file
```

We used the **cat** command to try to read the file:

```
cat /u/alease1/alease1file
```

**Result:** Permission was denied, as expected.

```
cat: /u/alease1/alease1file: EDC5111I Permission denied.
```

---

3. We issued the following RACF commands to give the ALEASE user ID READ access to the SUPERUSER.FILESYS profile:

```
PERMIT SUPERUSER.FILESYS CLASS(UNIXPRIV) ACCESS(READ) ID(ALEASE)
SETROPTS RACLIST(UNIXPRIV) REFRESH
```

---

4. We again tried to read the file from the ALEASE user ID:

```
cat /u/alease1/alease1file
```

**Result:** We were now able to read the file because of the permission to the SUPERUSER.FILESYS profile. The contents of the file were displayed:

```
hello1
```

---

5. We issued the following RACF commands to activate the SUPERUSER.FILESYS.ACLOVERRIDE profile:

```
RDEFINE UNIXPRIV SUPERUSER.FILESYS.ACLOVERRIDE UACC(NONE)
SETROPTS RACLIST(UNIXPRIV) REFRESH
```

---

6. We again tried to read the file (using the **cat** command).

**Result:** We were still able to successfully read the file. The override profile has no effect because there is no entry in the file's ACL to prevent user ALEASE from accessing the file. Thus, because user ALEASE has access to SUPERUSER.FILESYS, read access to the file is successful.

---

7. From a superuser ID, we set the ACLs for the alease1file file to explicitly deny all access to user ALEASE:

```
setfacl -vs u::rwx,g::r-x,o::---,u:alease:--- /u/alease1/alease1file
```

**Result:** The display from **getfacl** looks like this:

```
#file: /u/alease1/alease1file
#owner: ALEASE1
#group: EMPLOYEE
```

```
user::rwx
group::r-x
other:---
user:ALEASE:---
```

---

8. We issued the RACF command to activate ACL checking:

```
SETROPTS CLASSACT(FSSEC)
```

---

At this point, the following are in effect:

- User ALEASE is permitted to SUPERUSER.FILESYS.
- The SUPERUSER.FILESYS.ACLOVERRIDE profile is active—ACL authorizations override SUPERUSER.FILESYS authority.
- The FSSEC class is active—ACL checking is active.

9. We again attempted to read the alease1file file:

```
cat /u/alease1/alease1file
```

**Result:** As expected, permission is denied. User ALEASE cannot access the file because there is an entry in the access ACL prohibiting that user and the SUPERUSER.FILESYS.ACLOVERRIDE profile is enforcing the ACL authorization, overriding the user's access to SUPERUSER.FILESYS.

```
cat: /u/alease1/alease1file: EDC5111I Permission denied.
```

---

This concludes the example of using the SUPERUSER.FILESYS.ACLOVERRIDE profile.

## Using ISHELL with ACLs

You can use the following ISHELL panels to work with ACLs:

- Directory List — indicates when ACLs exist for a file or directory.
- Display File Attributes — includes ACL data.
- Edit Attributes — includes new support to edit the file access ACL and, for directories, the directory default ACL and file default ACL.
- Edit Access Control List — includes new support to add, delete, change, copy, or replace ACLs.

### Example of the Directory List and Display Attributes panels

When displaying a directory, a plus sign (+) in the **Perm** field indicates that extended ACL entries exist.

**Note:** To see the permissions, ensure that the display options are set to display the permissions field (**Options** → **1. Directory list...** → **/ Permissions (4 columns)**).

To display the attributes, from the Directory List Panel, type the letter a on the line and press Enter.

**Example:**

```

EUID=0 /u/alease1/
Type Perm Changed-EST5EDT Owner -----Size Filename Row 41 of 53
...
a Dir +700 yyyy-mm-dd hh:mm ALEASE1 8192 acldir
...
a File +700 yyyy-mm-dd hh:mm ALEASE1 0 setfac1_a
...

```

On the Display Attributes panel, a "1" appears in the **Access control list** field if an access ACL exists. For directories, a "1" in the **Directory default ACL** or **File default ACL** fields indicates the presence of those types of ACLs.

**Example:** The following is an example of the attribute display for the setfac1\_a file:

```

Pathname : /u/alease1/setfac1_a
...
File type : Regular file
Permissions : 700
Access control list . : 1

```

**Example:** The following is an example of the attribute display for the acldir directory:

```

Pathname : /u/alease1/acldir
...
Access control list . : 0
...
Directory default ACL : 1
File default ACL . . : 1

```

### Example of the Edit attributes menu

To edit ACLs, select the **Edit** pulldown menu on the Display File Attributes panel. On the **Edit** menu, options 8, 9, and 10 allow you to edit ACLs (options 9 and 10 only apply to directories).

**Example:** The following is an example of the **Edit** menu:

```

Edit Help
_
1. Mode fields...
2. Owning user...
3. Owning group...
4. User auditing...
5. Auditor auditing...
6. File format...
7. Extended attributes...
8. Access control list...
9. Directory default ACL...
10. File default ACL...

```

### Example of the Access Control List panel

From the **Edit** pulldown menu on the **Display File Attributes** panel, you can edit access ACLs (option 8), directory default ACLs (option 9), and file default ACLs (option 10).

The following options are available on the Access Control List panel:

- Use the **Add group** option to add an ACL entry for a group. The ACL must not already contain an entry for that group.

- Use the **Add user** option to add an ACL entry for a user. The ACL must not already contain an entry for that user.
- Use the **Copy** option to copy ACL entries from another file. Entries are ignored for any user or group that has an existing entry in the ACL.
- Use the **Replace** option to replace the ACL from the ACL of another file. All entries, if any, in the current ACL are deleted.

**Example:** The following is an example of the Access Control List panel for access ACLs (**Edit → 8. Access control list...**):

```

Access Control List: Access Row 1 to 2 of 2

Type over read, write or execute permissions to make a change.
Clear the value to reset it, anything else will set it.
To delete, place a D in the S column for an entry or use command D * for
all entries. Use commands SORT ID or SORT NAME to reorder the table.

Option: _ 1. Add group 2. Add user 3. Copy 4. Replace

S ID Name Read Write Execute Type
- 0 SYS1 R - X Group
- 9000 ALEASE9 R W X User
***** Bottom of data *****

```

**Example:** The following is an example of the Access Control List panel for directory default ACLs (**Edit → 9. Directory default ACL...**):

```

Access Control List: Directory Default Row 1 to 4 of 4

Type over read, write or execute permissions to make a change.
Clear the value to reset it, anything else will set it.
To delete, place a D in the S column for an entry or use command D * for
all entries. Use commands SORT ID or SORT NAME to reorder the table.

Option: _ 1. Add group 2. Add user 3. Copy 4. Replace

S ID Name Read Write Execute Type
- 0 sys1 - - - Group
- 2 BIN R - X Group
- 13456712 ALEASE R W X User
- 9000 ALEASE9 R W X User
***** Bottom of data *****

```

**Example:** The following is an example of the Access Control List panel for file default ACLs (**Edit → 10. File default ACL...**):

Access Control List: File Default

Row 1 to 2 of 2

Type over read, write or execute permissions to make a change.  
 Clear the value to reset it, anything else will set it.  
 To delete, place a D in the S column for an entry or use command D \* for  
 all entries. Use commands SORT ID or SORT NAME to reorder the table.

Option:    \_   1. Add group   2. Add user   3. Copy       4. Replace

| S | ID       | Name   | Read | Write | Execute | Type  |
|---|----------|--------|------|-------|---------|-------|
| _ | 0        | sys1   | R    | W     | X       | Group |
| _ | 13456712 | ALEASE | R    | _     | X       | User  |

\*\*\*\*\* Bottom of data \*\*\*\*\*

## Managing a hierarchical file system (HFS)

We covered our strategy and management of the hierarchical file system (HFS) in our December 2001 edition.

## Managing a zSeries file system (zFS)

The z/OS Distributed File Service zSeries File System (zFS) is a UNIX file system that can be used in addition to the Hierarchical File System (HFS). zFS does not replace HFS, but is complementary: HFS is still required for z/OS installation and the root file system must be HFS. See “Managing a hierarchical file system (HFS)” for information about our HFS environment.

We used the *z/OS Distributed File Service zSeries File System Administration*, SC24-5989 book to set up zFS.

## zFS terminology

The data set which contains a zFS filesystem is called a **zFS aggregate**. A zFS aggregate is a Virtual Storage Access Method (VSAM) Linear Data Set (LDS). A zFS aggregate can contain one or more zFS filesystems.

A zFS aggregate that contains only a single zFS filesystem is called a **compatibility mode aggregate**. Compatibility mode aggregates are similar to HFS in that you have one filesystem per data set. Conversely, a zFS aggregate that contains two or more zFS filesystems are called **multi-file system aggregates**. In our environment, we implemented only compatibility mode aggregates. We did this because for z/OS V1R2, zFS multi-file system aggregates are not supported in a shared HFS environment. Since we run a 14–system parallel sysplex, we needed to stick to compatibility mode aggregates where we can use the shared HFS capability we’ve set up.

## Setting up zFS

We did the following to set up zFS in our environment:

**Crating a JCL procedure for the zFS started task:** We created the following JCL procedure for the ZFS started task in SYS1.PROCLIB by copying the sample procedure for from data set hlq.SIOEPROC:

## Managing an zFS

```
//ZFS PROC REGSIZE=0M
//ZFSGO EXEC PGM=BPXVCLNY,REGION=®SIZE,TIME=1440
//*STEPLIB DD DISP=SHR,DSN=HLQ.SIOELMOD
//IOEZPRM DD DSN=SYS1.&SYSNAME..ZFS.PARMLIB(IOEFSPRM),DISP=SHR
//* PEND
```

The IOEZPRM DD statement identifies the zFS parameter data set, IOEFSPRM. We'll show how we defined the IOEFSPRM data set in one of the steps below.

**Issued RACF commands to define zFS to the started class:** We issued the following RACF commands to define zFS to the started class for the procedure and then refresh the started class:

```
RALT STARTED ZFS.** STDATA(USER(DFS))
SETR RACLIST(STARTED) REFRESH
```

**Created the IOEFSPRM zFS parameter data set for each system:** For each system where we're defining zFS, we created a data set, IOEFSPRM, to contain the zFS parameters. We've already specified the IOEZPRM DD statement in the ZFS PROC to contain the name of the IOEFSPRM member. Note that the IOEFSPRM member is optional; you only need it if you want to override the defaults for the zFS parameters. We're using it to specify trace and debug data sets for each system. The following examples show how we created and customize the IOEFSPRM member for one of our systems, Z0.

- The following example shows the contents of the IOEFSPRM member for system Z0 in SYS1.Z0.ZFS.PARMLIB:

```
user_cache_size=256m
debug_setting_dsn=sys1.z0.zfs.debug
ioedebug=(c=vm,p=m,l=127)
trace_dsn=sys1.z0.zfs.trace
trace_table_size=32m
```

- The following example shows the attributes of the trace and debug data data sets we defined:

```
sys1.z0.zfs.parmlib -po- recfm=fb,space=trk(1,1),lrecl=80,blksize=6160
sys1.z0.zfs.debug -ps- recfm=fb,space=trk(1,1),lrecl=80,blksize=27920
sys1.z0.zfs.trace -pdse- recfm=vb,space=cyl(100,20),lrecl=133,blksize=27998
```

**Created a BPXPRM00 entry for zFS:** We added the following statement to our BPXPRM00 parmlib member in data set SYS1.PETR12.PARMLIB:

```
FILESYSTYPE TYPE(ZFS) ENTRYPOINT(IOEFSCM) ASNAME(ZFS)
```

**Defined and formatted a new VSAM LDS for zFS:** We used the following JCL to create a new VSAM LDS for our zFS compatibility mode aggregate:

```
//ZFSDEFA JOB MSGCLASS=H,MSGLEVEL=(1,1),
// CLASS=A
//*MAIN SYSTEM=Z0
/*JOBPARM SYSAFF=Z0
//*****
//*
//* FUNCTION: DEFINE A ZFS AGGREGATE
//*
//*****
//DEFINE EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//AMSDUMP DD SYSOUT=*
//DASDO DD DISP=OLD,UNIT=3390,VOL=SER=SS0048
//SYSIN DD *
DEFINE CLUSTER (NAME(OMVSSPN.U2.AGGR.U078001) VOLUMES(SS0048) -
LINEAR CYL(15 0) SHAREOPTIONS(2) STORCLAS(SMSOE))
//*****
//*
```

```

// * FUNCTION: FORMAT VSAM LINEAR DATASET AS A HFS COMPAT AGGREGATE
// *
// *****
//NEWZFSA EXEC PGM=IOEAGFMT,REGION=0M,
// PARM=(' -aggregate OMVSSPN.U2.AGGR.U078001 -compat')
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//STDOUT DD SYSOUT=*
//STDERR DD SYSOUT=*
//CEEDUMP DD SYSOUT=*
// *

```

**Mounting zFS file systems:** After we set up and started our zFS task, we mounted our zFS aggregate using a TSO/E MOUNT command. You can use the ISHELL (ISH) command to mount zFS filesystems. As of z/OS V1R3, you can also use the MOUNT parameter in parmlib member BPXPRMxx to mount zFS filesystems at IPL time.

**Attaching our compatibility mode aggregates to our systems:** Compatibility mode aggregates are automatically attached to the system they are created on at the time the filesystem is mounted. That meant that we did not have to manually attach our zFS aggregates to their systems using the zfsadm attach command or a define\_aggr statement in the IOEFSPRM file. Using one of these manual attach methods is required for multi-file system aggregates.

**zFS automoves:** In a zFS environment, whether zFS filesystems get automoved depends on whether you're running in a mixed sysplex or in a sysplex with all members at z/OS V1R2 or higher. zFS aggregates have to be mounted on a z/OS V1R2 or higher level system with the zFS started task running, but they can coexist in a mixed level (V1R2 and lower) sysplex.

- In a mixed sysplex, zFS aggregates can be automoved in case of system failure to another z/OS V1R2 or higher system with zFS running, depending on the situation:
  - If you either issue the VARY command to vary a system out of the sysplex or the F BPXOINIT SHUTDOWN=FILESYS command to unmount or move filesystems mounted with the automove parameter on the system, ownership of the zFS aggregates are automoved to another z/OS V1R2 or higher system where zFS is running (if one exists) and all the mounts will stay active.
  - If you manually stop or cancel zFS, or if it abends, the zFS aggregates are not automoved. In this case, all the zFS filesystems in the sysplex will be unmounted.
- In a sysplex where all members are at z/OS V1R2 or higher with zFS running, filesystems mounted as automovable can be moved to another system in the sysplex where zFS subsystem is active. This will occur in one of the following situations:
  1. You vary the owning system out of the sysplex
  2. You use command F BPXOINIT,SHUTDOWN=FILESYS
  3. You shutdown the zFS subsystem on the owning system. If zFS is not running on any other system in the sysplex then the zFS filesystems will be unmounted and not automoved.

**Using zFS in a mixed sysplex:** If you are running in a mixed sysplex, keep in mind that non-zFS systems cannot view any zFS directory structures or zFS filesystem contents. A non-zFS system can be either of the following:

- A system at a z/OS V1R1 or lower release level
- A system at a z/OS V1R2 or higher release level, but not running zFS

## Managing an zFS

Using zFS in a mixed sysplex can lead to some confusion, if, like we do, you sometimes have a zFS and an HFS sharing the same mount point. The figures below illustrate the situation.

In Figure 11, systems in a mixed sysplex view the contents of directory /pgm1 in HFS /web on system Z0. In this picture, systems are at the following levels:

- J80 is at the z/OS V1R1 level
- JH0 is z/OS V1R2 not running zFS
- Z0 is V1R2 running zFS

So far, so good; all three systems see the HFS contents of /web/pgm1:

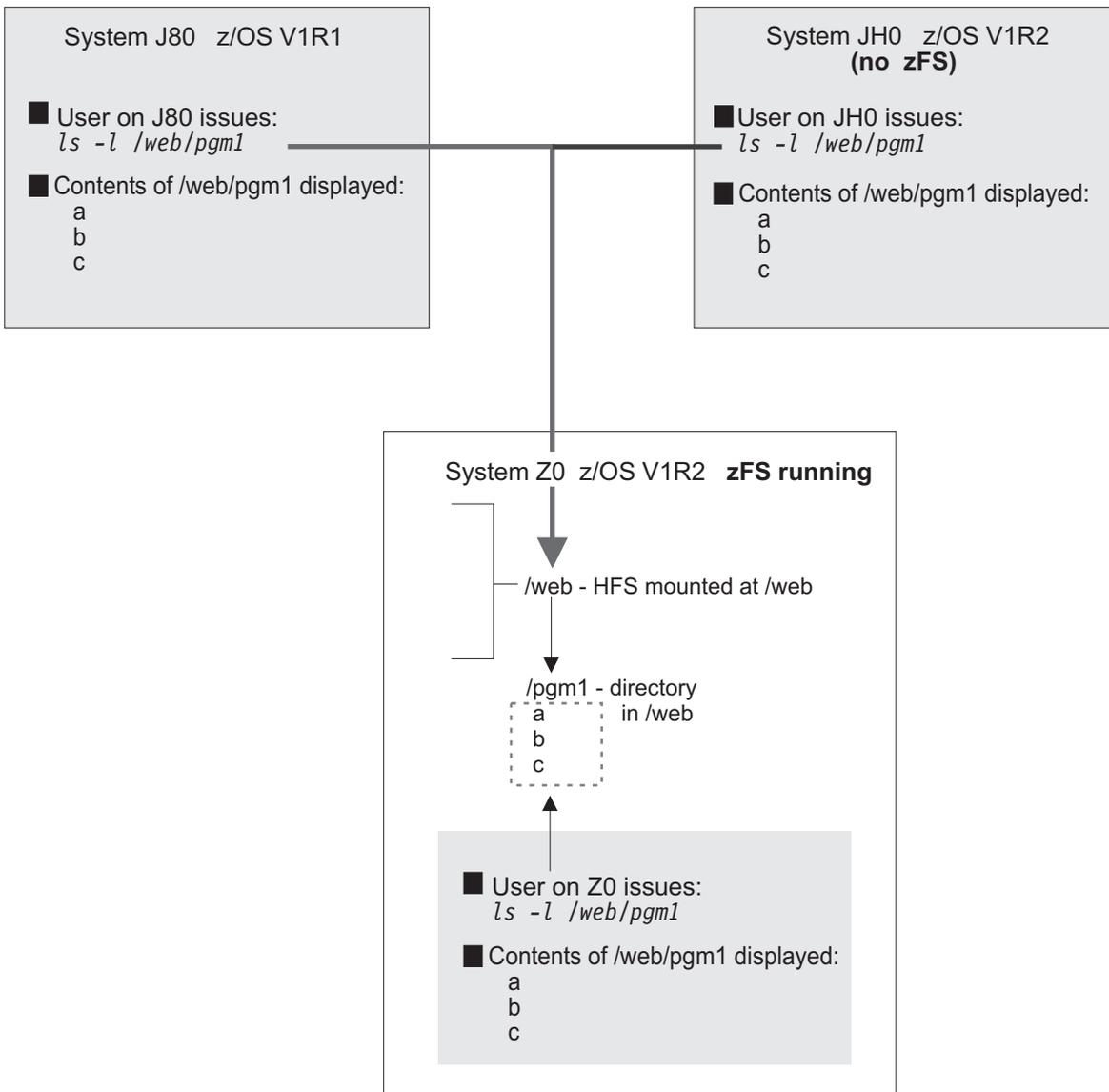


Figure 11. Mixed sysplex viewing HFS filesystem contents

Now let's say we mount a zFS filesystem at /web/pgm1 on system Z0. At the same time, we also start running zFS on system JH0. This changes things, because both JH0, now a zFS system, and system Z0 can view our new zFS. System J80, however, is still a non-zFS system and can only view the HFS filesystem contents.

This means that users on the different systems in our mixed sysplex will see different contents of /web/pgm1. Because users might not know whether their system is zFS or non-zFS, they may be seeing different contents of /web/pgm1 without knowing it. In Figure 12, J80 sees the HFS contents of /web/pgm1, while systems JH0 and Z0 see the zFS contents:

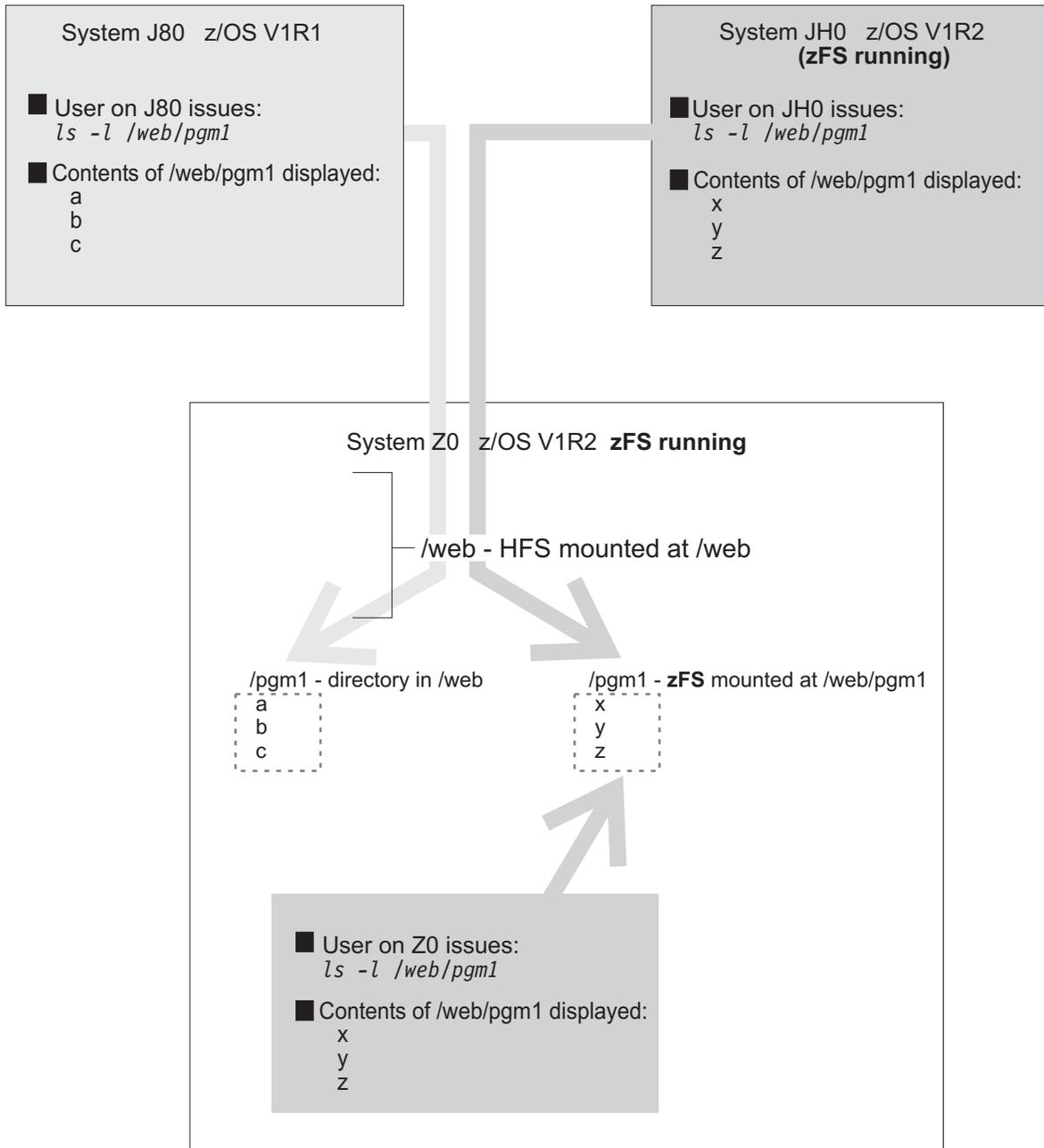


Figure 12. Mixed sysplex viewing HFS and zFS filesystem contents

Note that because Z0 is a zFS system, it will **only** see the contents of the zFS mounted at /web/pgm1, even though it owns the /web HFS.



---

## Chapter 9. Using IBM HTTP Server

This chapter describes our experiences with IBM HTTP Server V5.3 during our testing of z/OS V1R3.

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](http://www.ibm.com/software/webservers/appserv/troubleshooter.html)).

---

### Protecting IBM HTTP Server resources with LDAP

In our December 2000 edition, we described how we set up protection for IBM HTTP Server resources with LDAP using the TCP transport. We now build upon that experience and describe how we set up protection for the HTTP server with LDAP using the SSL transport. The work we describe in this section is based on our previous work in setting up LDAP protection with the TCP transport (see the section on “Using LDAP to protect IBM HTTP Server resources” in the IBM HTTP Server chapter of our December 2000 edition).

**Required service:** In order for LDAP protection with the SSL transport to work, we found that we needed to install the following service:

- IBM HTTP Server APAR PQ52592
- LDAP Server APAR OW51259

**Applicable documentation:** We used information from the following sources to help us appropriately customize IBM HTTP Server and z/OS Security Server LDAP Server to enable LDAP protection:

- *z/OS HTTP Server Planning, Installing, and Using*, SC34-4826
- *z/OS Security Server LDAP Server Administration and Use*, SC24-5923

### Customizing the HTTP server configuration file

The following is a portion of our `httpd.conf` configuration file showing the changes we made for LDAP protection using the SSL transport:

```
LDAPInfo LDAPssl {
 Host LDAP_server_IP_address
 Transport SSL
 SearchTimeout 2 minutes
 ClientAuthType BasicIfNoCert
 ServerAuthType Basic
 ServerDN "cn=LDAP Administrator, o=Your Company"
 ServerPasswordStashFile /dir_path/LDAPStash
 KeyFilePasswordStashFile /dir_path/LDAPsslStash
 UserSearchBase "o=Your Company"
 UserNameFilter "(&(objectclass=organizationalPerson)(cn=%v1 %v2))"
 UserCertFilter "(&(objectclass=organizationalPerson)(cn=%v1 %v2))"
 GroupNameFilter "(objectclass=groupOfNames)"
 KeyFileName /etc/ldap/keys/mykey.kdb
 KeyLabel "cert"
}

Protection LDAPssl {
 ServerId LDAPssl
 UserId %%SERVER%%
 PasswdFile "%%LDAP:LDAPssl%%"
 GroupFile "%%LDAP:LDAPssl%%"
}
```

## IBM HTTP Server

```
 Mask "cn=Bowling team, ou=Groups, o=Your Company"
 }
Protect /html//LDAPssl.html LDAPssl
```

Note the following information about two of the subdirectives under the LDAPInfo directive shown above:

### KeyFilePasswordStashFile

The KeyFilePasswordStashFile subdirective is required; however, the HTTP Server documentation did not point this out. The fix for IBM HTTP Server APAR PQ52592 includes updates to the HTTP Server documentation to describe the KeyFilePasswordStashFile subdirective. (These updates are now available in the latest edition of the documentation available on the IBM HTTP Server library Web site at [www.ibm.com/software/websphere/httpservers/library.html](http://www.ibm.com/software/websphere/httpservers/library.html).) This subdirective specifies the name of the stash file created to hold the password to the key file. The **htadm** command is used to create this stash file. A stash file created by `ikeyman/gskkyman` cannot be used for this purpose.

### KeyFileName

The HTTP Server documentation said that the default value for the KeyFileName subdirective is the file specified by the KeyFile directive. This seemed to imply that the KeyFileName subdirective *could* specify a different file than the KeyFile directive. However, we found this not to be the case; the value of the KeyFileName subdirective *must* match that of the KeyFile directive. The fix for APAR PQ52592 also updates the HTTP Server documentation to say that if SSL is used both for client/browser connections to the HTTP server and for the HTTP server's connection to an LDAP server, then the KeyFileName subdirective must specify the same key file that is named on the KeyFile directive.

## Customizing the LDAP server configuration file

Through previous work, our LDAP server was already enabled for SSL (see *z/OS Security Server LDAP Server Administration and Use* for details about enabling the LDAP server for SSL). However, we did need to add the `sslCipherSpecs` directive to the `slapd.conf` configuration file to get the communications to work. The value specified by the `sslCipherSpecs` directive represents a bit map that indicates which SSL cipher specifications will be accepted from clients. Clients that support any of the specified cipher specifications will be able to establish an SSL connection with the server.

We specified the `sslCipherSpecs` directive in the `slapd.conf` file as follows:

```
sslCipherSpecs 15104
```

## Verifying the LDAP protection for HTTP server resources

To verify that LDAP protection is working properly, we attempted to access our protected HTML document, `LDAPssl.html`.

### Steps to verify the LDAP protection for our protected resource

We did the following:

1. We entered the following URL in our Web browser:

```
http://http_server_ip_address/LDAPssl.html
```

2. A dialog box appeared requesting a user name and password. We entered:

**User Name:** Eddie Catu

**Password:** *eddie's\_password*

---

3. Our LDAPssl.html document then appeared in the browser window.

---

We were able to successfully access our protected HTTP server resource using LDAP protection with the SSL transport.



---

## Chapter 10. 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”
- “Enabling native authentication” on page 126
- “Setting up a referral between LDAP databases” on page 132
- “Enabling LDAP for Kerberos authentication” on page 134
- “Resolving a problem with malformed distinguished names in directory entries” on page 137

---

### Overview of our LDAP configuration

We have a multiplatform LDAP configuration and we use both replication and referral. Figure 13 shows a high-level view of our LDAP multiplatform configuration:

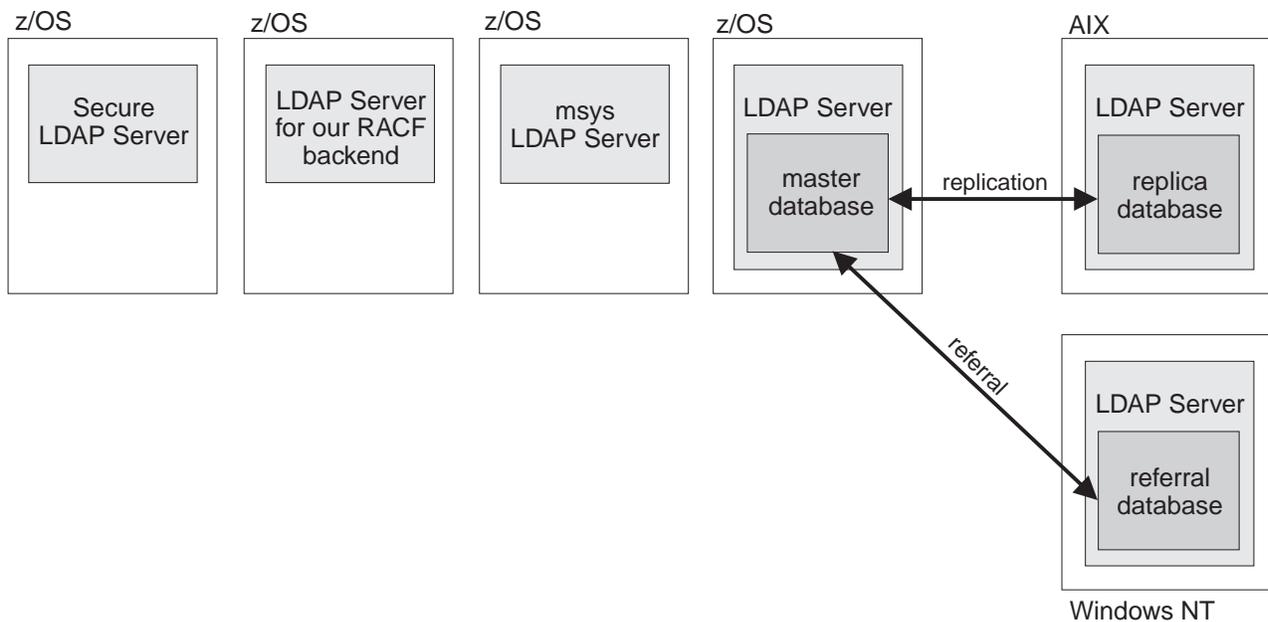


Figure 13. Overview of our LDAP configuration

Our LDAP environment includes the following features:

- **Secure LDAP server:** We set up one LDAP server for SSL secure transactions. The server is bound to the same DB2 database as our master server, so all of the entries are identical. This server has a TDBM backend and listens 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.
- **msys LDAP server:** We have an LDAP server for the z/OS Managed System Infrastructure, msys.
- **Master LDAP server:** Our master server on z/OS has a DB2 backend database, TDBM.

## LDAP Server

- **LDAP replica server on AIX:** We manage our AIX replica server through a Web-based management tool at:

`http://AIX_server_IP_address/ldap/index.html`

Our AIX server has a DB2-based backend and was configured through the Web-based management tool to be a replica of the master LDAP server on z/OS.

- **LDAP referral server on Windows NT:** We manage our Windows NT referral server using the Web-based management tool at:

`http://NT_server_IP_address/ldap/index.html`

Our Windows LDAP server has a DB2-based backend and has a general referral pointing to our master LDAP server on z/OS. This allows the user to run the **ldapsearch** command from the Windows NT LDAP server to search for entries in both the Windows NT Server directory and the master server directory. The **ldapsearch** command returns all the matching entries from both directories.

---

## Enabling native authentication

This section describes what we did to enable native authentication for LDAP.

The z/OS LDAP Server allows clients to bind to entries in a TDBM backend by using the system for verifying the authentication attempt. Using native authentication, a client performs a simple bind supplying an LDAP DN of an entry in a TDBM backend along with a Security Server (RACF) password. The Security Server then performs the password authentication.

We followed the instructions in *z/OS Security Server LDAP Server Administration and Use*, SC24-5923, to enable native authentication.

## Updating the schema for native authentication

We updated our directory schema with the NativeAuthentication.ldif file, as follows.

### Steps to update our schema

1. We issued the following command to copy the NativeAuthentication.ldif file to our ldap directory:

```
cp /usr/lpp/ldap/etc/NativeAuthentication.ldif /etc/ldap
```

2. We edited our new copy of the NativeAuthentication.ldif file, changed the suffix in the file to o=Your Company and saved the file.

3. We issued the following **ldapmodify** command to update the schema:

```
ldapmodify -h ip_address -D "cn=LDAP Administrator, o=Your Company, c=US" -w secret
-f /etc/ldap/NativeAuthentication.ldif
```

**Result:** We received the following response to the **ldapmodify** command:  
modifying entry cn=schema, o=Your Company

---

This completes the procedure to update the schema.

## Learning from our mistakes

We discovered that we made a few mistakes during our initial testing of native authentication. The errors were not apparent to us at first, so we'd like to summarize them here in the hope that others might benefit from them. We have also been made aware that the next release of the LDAP Server documentation will provide additional information to help clarify some of these issues we discuss below.

In order to quickly implement native authentication, we used existing LDAP entries and RACF user IDs. This turned out to be a bad idea because the user IDs and passwords were the same. This also masked a problem later when we tried to verify our work by using the HTTP server to access a protected Web resource whose protection scheme was enabled for native authentication. When the native authentication attempt failed, the LDAP server defaulted to using the same user ID and password from its own database instead of the user ID and password from RACF. Thus, on the surface, the native authentication function appeared to be working.

We took actions in the following areas to complete our testing and ensure that native authentication was working properly:

**Updating the `slapd.conf` file:** We made an error by not enclosing the value of the `nativeAuthSubtree` attribute in double quotes. We originally specified the attribute as follows:

```
nativeAuthSubtree ou=Insko,o=Your Company
```

Because the attribute value contained a space character but the value was not contained in quotes, the search for the entry used a value of `ou=Insko,o=Your` which is not valid. To fix this, we put double quotes around the attribute value, as follows:

```
nativeAuthSubtree "ou=Insko,o=Your Company"
```

**Ensuring RACF user IDs have OMVS segments:** The RACF user ID that is specified in an `ibm-nativeid` attribute must have an OMVS segment defined.

**Changing the RACF password:** In order to use the `ldapmodify` command to change a native (RACF) password, the LDAP ACLs must be properly set to allow an update of the `userpassword` attribute.

**Performing additional validation:** In addition to using the HTTP server to access a protected Web resource using native authentication, we use the `ldapsearch` command to validate that native authentication is working. The following is an example of this command:

```
ldapsearch -h ip_address -D "cn=c00002,ou=Insko,o=Your Company" -w racf_pw
-b "cn=c00002,ou=Insko,o=Your Company" "(objectclass=*)"
```

**Finding out more information:** Refer to information APAR II13026 for more information about LDAP Server at the z/OS V1R2 level.

## Updating the `slapd.conf` configuration file

As *z/OS Security Server LDAP Server Administration and Use* describes, there are many different configuration options for native authentication. We decided to restrict participation in native authentication to our Insko entries. Thus, we added the following native authentication options to our `/etc/ldap/slapd.conf` configuration file:

## LDAP Server

```
useNativeAuth selected
nativeUpdateAllowed yes
nativeAuthSubtree "ou=Insco,o=Your Company"
```

Based on the above option settings, only those entries in the specified subtree (ou=Insco,o=Your Company) which contain the **ibm-nativeId** attribute will be subject to native authentication.

## Modifying directory entries to perform native authentication

The options we specified in our slapd.conf file require that entries contain the **ibm-nativeId** attribute in order to participate in native authentication. Therefore, we did the following to modify one of our existing Insco entries, C00002, to enable it for native authentication.

**Note:** We have an existing, corresponding C00002 user ID and password defined in RACF. This user ID must also have an OMVS segment defined.

### Steps to modify one of our entries

1. We created a C00002.nativeauth file containing the following data to add the **ibm-nativeId** attribute type:

```
dn: cn=c00002, ou=Insco, o=Your Company
changetype: modify
add: x
ibm-nativeId: C00002
-
add: x
objectclass: ibm-nativeAuthentication
-
```

2. We issued the following **ldapmodify** command to update the C00002 ID:

```
ldapmodify -h ip_address -D "cn=LDAP Administrator, o=Your Company" -w secret
-f /etc/ldap/C00002.nativeauth
```

**Result:** The command completed successfully and we got the following response:

```
modifying entry cn=c00002, ou=Insco, o=Your Company
```

#### A mistake we made

The first time we tried this, we only added the **ibm-nativeId** attribute in step 1; we forgot about the **ibm-nativeAuthentication** object class. As a result, when we issued the **ldapmodify** command, we got the following error message:

```
modifying entry cn=c00002, ou=Insco, o=Your Company
ldap_modify: Object class violation
ldap_modify: additional info: R001030 Entry contained attribute type not
allowed by schema: ibm-nativeid. (schemaimpl.c|1.75|2004)
```

After we went back and added the native authentication object class, we reissued the **ldapmodify** command and it completed successfully, as shown above.

This completes the procedure to modify the entry for our C00002 ID. We then took the following steps to verify the changes that we just made.

### Steps to verify our changes

1. We issued the following **ldapsearch** command to verify that our C00002 ID had the newly-added **ibm-nativeId** attribute and **ibm-nativeAuthentication** object class:

```
ldapsearch -h ip_address -b "cn=C00002, ou=Insko, o=Your Company" "(objectclass=*)"
```

#### Result:

```
cn=c00002, ou=Insko, o=Your Company
objectclass=organizationalPerson
objectclass=PERSON
objectclass=TOP
objectclass=InskoPerson
objectclass=ibm-nativeauthentication
cn=c00002
sn=c00002
ou=Insko
inscobadlogoncount=1
inscoacctlocked=FALSE
```

We noticed that, in the above response, the **ibm-nativeAuthentication** object class appears, but the **ibm-nativeId** attribute does not appear.

2. We decided to try searching on just the **ibm-nativeId** attribute, as follows:

```
ldapsearch -h ip_address -b "cn=C00002, ou=Insko, o=Your Company" "(objectclass=*)" ibm-nativeId
```

#### Result:

```
cn=c00002, ou=Insko, o=Your Company
no data...
```

We learned that the ldap search was being performed with “anonymous” access and that such access is not permitted to retrieve the **ibm-nativeId** attribute.

3. We performed the search again using LDAP Administrator access, as follows:

```
ldapsearch -h ip_address -D "cn=LDAP Administrator, o=Your Company" -w secret
-b "cn=C00002, ou=Insko, o=Your Company" "(objectclass=*)" ibm-nativeId
```

#### Result:

```
cn=c00002, ou=Insko, o=Your Company
ibm-nativeId=C00002
```

We see that the C00002 entry does indeed contain the **ibm-nativeId** attribute that we added earlier.

4. We issued the following command to test the native authentication function to retrieve data:

```
ldapsearch -h ip_address -D "cn=c00002,ou=Insko,o=Your Company" -w racf_pw
-b "cn=c00002,ou=Insko,o=Your Company" "(objectclass=*)"
```

This completes the verification of our changes to enable native authentication.

### Enabling the IBM HTTP Server to exploit LDAP native authentication

We set up protection on one of our IBM HTTP Servers to make use of LDAP native authentication. We protected a resource (an HTML document named LDAPnative.html) and then tried to access that resource using the C00002 user ID that we enabled earlier for native authentication.

#### Overview of HTTP server protection and LDAP native authentication

Figure 14 shows an overview of the process to access a protected HTML document using native authentication.

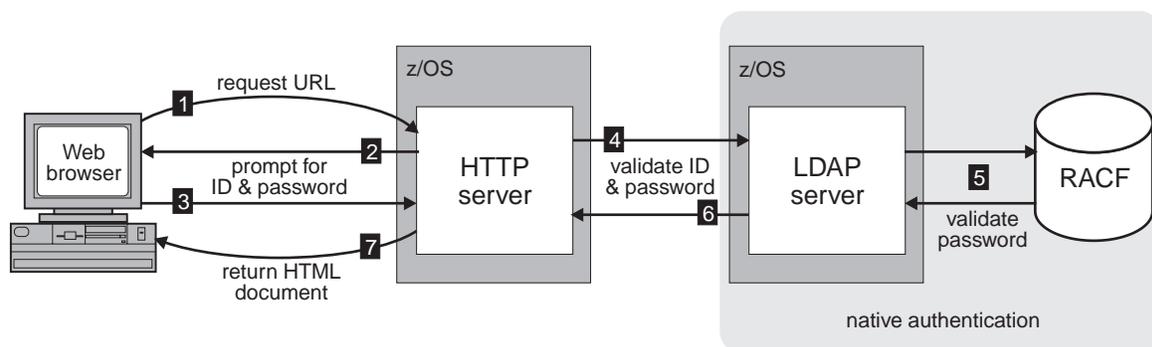


Figure 14. Overview of HTTP server protection and LDAP native authentication

The following steps describe the process shown in Figure 14:

- 1** The Web browser sends a request containing the URL of our HTML document to the HTTP server.
- 2** The HTTP server determines that the HTML document is protected and sends an authentication challenge back to the browser. The browser displays a prompt for a user ID and password.
- 3** The user at the browser types the user ID and password which the browser sends back to the HTTP server.
- 4** The HTTP server searches LDAP for the user ID, receives the returned DN, and issues an LDAP bind request using the DN and password.
- 5** The LDAP server determines that the entry for the DN is subject to native authentication, retrieves the **ibm-nativeID**, and uses that ID to verify the password with the security server (RACF).
- 6** The LDAP server returns a successful bind to the HTTP server.
- 7** The HTTP server sends the protected HTML document to the browser for display.

#### Steps to enable our IBM HTTP Server to exploit native authentication

The following steps describe how we enabled our IBM HTTP Server to exploit native authentication:

1. We added the following directives to our httpd.conf file to specify our LDAP server information, define our protection setup, and apply the protection setup to our LDAPnative.html document:

```
LDAPInfo LDAPnative {
 Host ip_address
 Transport TCP
 SearchTimeout 2 minutes
 ClientAuthType Basic
 ServerAuthType Basic
 ServerDN "cn=LDAP Administrator, o=Your Company"
 ServerPasswordStashFile /JF0/dgw_pet/LDAPStash
 UserSearchBase "o=Your Company"
 UserNameFilter "&(objectclass=organizationalPerson)(cn=%v1)"
 GroupNameFilter "(objectclass=groupOfNames)"
}

Protection LDAPnative {
 ServerId LDAPnative
 UserId %%SERVER%%
 PasswdFile "%%LDAP:LDAPnative%%"
 GroupFile "%%LDAP:LDAPnative%%"
 Mask "cn=Clients, ou=Insko, o=Your Company"
}

Protect /html/LDAPnative.html LDAPnative
```

2. We stopped and restarted the HTTP server to activate the configuration changes.
3. On a workstation, we entered the URL for our protected HTML document (LDAPnative.html) into the Web browser:  
http://xxxxx.ibm.com/html/LDAPnative.html

4. As expected, the Web browser prompted for a user ID and password. We entered the following:

**User ID:** C00002

This is the user ID that we specified in the **ibm-nativeid** attribute for our LDAP entry, above. We have a corresponding C00002 user ID defined in RACF.

**Password:** The RACF password for user C00002

5. The browser successfully displayed the LDAPnative.html document.

Through the LDAP native authentication function, the HTTP server successfully authenticated our request for the protected Web resource.

### Setting up a referral between LDAP databases

Referral processing allows a user to link multiple LDAP directories on multiple platforms. This type of processing allows a user to issue a search request against one LDAP server and, if the requested entry is not found, continue searching another server to locate the requested entry.

To test referral processing, we set up a referral from our LDAP server on z/OS to an LDAP server on Windows NT.

### Setting up an LDAP server on Windows NT

Our first task was to set up a new LDAP server image on an existing Windows NT server in our test environment. To do this, we installed the following software products:

- IBM SecureWay Directory V3.2.2 for Windows NT
- GSKit V5 SSL
- IBM DB2 UDB V7.2
- IBM HTTP Server V1.3.19

#### Steps for setting up an LDAP server on Windows NT

We performed the following steps to set up an LDAP server on Windows NT:

1. We installed IBM SecureWay Directory V3.2.2 for Windows NT along with the other supporting products listed above.

---
2. After we successfully installed SecureWay Directory, we used its Directory Management Tool to define a new "o=IBM, c=US" suffix, as follows:
  - a. In the Directory Management Tool, we went to the **Server Administration** section.
  - b. The tool requested the Web server address and an optional port number. We entered the requested information.  
**Result:** The tool opens a Web-based interface to configure the LDAP server.
  - c. We logged in using the LDAP server administrator's user ID and password.
  - d. We chose **Settings** → **Suffixes**.
  - e. We entered our "o=IBM, c=US" suffix, then clicked **Update**.
  - f. We restarted the Windows NT system.

---
3. At the DB2 command prompt, we issued the following command to load the sample data file into the database:

```
bulkload -i filename
```

---
4. To verify that the data was successfully loaded and that LDAP was working properly, we issued the following search command:

```
ldapsearch -h NT_server_ip_address -b "o=IBM, c=US" "(objectclass=*)"
```

**Result:** All of the entries under the specified suffix appeared on the screen.

---

The LDAP server was now successfully running on Windows NT.

## Defining a referral in LDAP Server on z/OS

Referral processing allows you to define two different types of referrals:

- Default referrals that are defined in a referralObject in the slapd.conf configuration file (You can define multiple default referrals.)
- Referrals that are defined in a referralObject in a specific LDAP directory

We tested the latter type by defining a referral in a specific LDAP directory in our LDAP server on z/OS which points to our LDAP server on Windows NT.

### Steps for defining a referral in z/OS LDAP Server

We did the following to define a referral in our LDAP server on z/OS to our LDAP server on Windows NT:

1. We created an ldif file with an entry for "o=IBM, c=US" and a referral to our LDAP server on Windows NT. The contents of the ldif file are:

```
dn:o=IBM,c=US
objectClass:referral
ref:ldap://NT_server_ip_address/o=IBM,c=US
```

Note that the ref: statement points to the LDAP URL of the referred-to server which, in our case, is the LDAP server on Windows NT that we set up earlier.

2. To make the new entry work, we needed to add a suffix entry to the slapd.conf file for the LDAP server on z/OS. We added the suffix entry to the TDBM definitions section of the slapd.conf file, as follows:

```
#tdbm database definitions
#####
database tdbm GLDBTDBM
suffix "o=Your Company"
suffix "o=IBM, c=US"
```

3. We issued the following command to load the above ldif file into the LDAP server on z/OS:

```
ldapadd -h z/OS_LDAP_server_ip_address -D "cn=LDAP Administrator" -w xxxxx -f xxxx.ldif
```

4. We recycled the LDAP server on z/OS.

The referral from the LDAP server on z/OS to the LDAP server on Windows NT is now available.

## Testing the referral function

To test the referral processing, we issued the following search command to the LDAP server on z/OS:

```
ldapsearch -h z/OS_LDAP_server_ip_address -b "o=IBM, c=US" "(objectclass=*)"
```

When the command completed, it displayed on the screen a list of the entries that are members of the "o=IBM, c=US" group. These are the entries that are stored on the LDAP server on Windows NT, not on the LDAP server on z/OS.

## LDAP Server

To further prove this point, we disabled the LDAP server on Windows NT and repeated the above search command. This time the command response said that the entry on the z/OS LDAP server was a referral and that the referral server could not be contacted to retrieve the rest of the data. Thus, the referral from the z/OS LDAP server to the LDAP server on Windows NT was working correctly.

---

## Enabling LDAP for Kerberos authentication

z/OS LDAP Server allows clients to authenticate to the server by using IBM's Network Authentication Service, better known as Kerberos Version 5. This section describes our experiences with enabling the LDAP server for Kerberos authentication.

**Applicable documentation:** We used information from the following sources to help us set up Kerberos authentication for LDAP:

- *z/OS Security Server LDAP Server Administration and Use*, SC24-5923
- *z/OS SecureWay Security Server LDAP Client Programming*, SC24-5924
- *z/OS SecureWay Security Server Network Authentication Service Administration*, SC24-5926

## Experiences with enabling LDAP for Kerberos authentication

We followed the steps for setting up for Kerberos in the chapter on Kerberos authentication in *z/OS Security Server LDAP Server Administration and Use*. In general, we found some of the steps to be confusing, partly because we lacked a full understanding of Kerberos authentication and its terminology. This is where a little “homework” can really pay off—researching and understanding how Kerberos authentication works is critical for performing this task.

Another factor that seemed to contribute to some of the problems we experienced was that our Kerberos server and LDAP server were on different z/OS images in our sysplex. Although this should work fine, after experiencing some problems we decided to try to simplify matters and bring up a Kerberos server on the same image as the LDAP server.

The following sections describe our experiences in greater detail.

### Performing the setup steps

The following are some of our experiences with performing the Kerberos setup steps in *z/OS Security Server LDAP Server Administration and Use*.

**Using the ALTUSER command to create a Kerberos identity for the LDAP server:** The step to issue the ALTUSER command to create a Kerberos identity for the LDAP server user ID confused us because of the addition of the password. The following is an example of the command:

```
ALTUSER ldap_id PASSWORD(password) NOEXPIRED KERB(KERBNAME(LDAP/hostname))
```

By changing the RACF password, it updates or creates the Kerberos segment for the specified user ID in the RACF database. It also assigns the specified password to the Kerberos principal that is being created. You can see this later when we use the **kin**it command to verify our setup.

Note that the NOEXPIRED parameter is critical and must be specified. The initial documentation that we used did not specify this parameter. The NOEXPIRED parameter is needed to set the key version of the LDAP server's Kerberos key. You can find the key version by issuing the following RACF command:

```
LISTUSER ldap_id NORACF KERB
```

Also, earlier in *z/OS Security Server LDAP Server Administration and Use* under the section on defining the Kerberos identity in the chapter on setting up the user ID and security for the LDAP server, you can see the use of another ALTUSER command to remove the password from the user ID that starts the LDAP server, as follows:

```
ALTUSER ldap_id NOPASSWORD
```

We recommend issuing that command again here. This does not affect the Kerberos principal that was generated by the first ALTUSER command (above) nor its password. However, it does change the user ID that starts the LDAP server back to being a user ID that cannot login.

**Generating a keytab file for the LDAP server:** The documentation indicates that if the LDAP server and Kerberos server are on the same z/OS image, then you do not need to generate a keytab file for the LDAP server. Initially, we had these two servers on different z/OS images, so we executed the steps to generate a keytab file. If the **krbKeytab** directive in the slapd.conf file specifies a keytab file but the keytab file does not exist, Kerberos authentication fails with the following error message:

```
GLD0173E Server was unable to acquire Kerberos credentials.
```

Therefore, if you do not have a keytab file, then make sure that the **krbKeytab** directive is set to none. If you do use a keytab file, you can issue the following command to validate the results of the **keytab add** command:

```
keytab list
```

The response to the command should look something like this:

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

```
Principal: LDAP/hostname.xxxxx.ibm.com@xxxxx.IBM.COM
Key version: 1
Key type: 56-bit DES
Entry timestamp: yyyy/mm/dd-hh:mm:ss
```

```
Principal: LDAP/hostname.xxxxx.ibm.com@xxxxx.IBM.COM
Key version: 1
Key type: 56-bit DES using key derivation
Entry timestamp: yyyy/mm/dd-hh:mm:ss
```

```
Principal: LDAP/hostname.xxxxx.ibm.com@xxxxx.IBM.COM
Key version: 1
Key type: 168-bit DES using key derivation
Entry timestamp: yyyy/mm/dd-hh:mm:ss
```

**Verifying the Kerberos principal:** We issued the **kinit** command to verify the Kerberos principal that was created earlier as a result of the ALTUSER command:

```
kinit LDAP/hostname.xxxxx.ibm.com@xxxxx.IBM.COM
```

A password prompt appeared. We entered the password that we defined earlier on the ALTUSER command.

If everything was fine, we should've returned to the command prompt. However, it was at this point that we realized we had some problems with our Kerberos (that is, z/OS Security Server Network Authentication Service) setup. We first needed to remedy those problems before we could continue with enabling the LDAP server for

## LDAP Server

Kerberos authentication. For details, see Chapter 11, "Setting up Security Server Network Authentication Service" on page 139.

After we were certain that our Network Authentication Service setup was correct (by issuing the **kinit** command), we still had problems with the LDAP enablement of Kerberos authentication. We determined that there was a problem with our DNS and resolver setup because we kept getting our sysplex name added into our IP name. For example, where we would expect to see *hostname.xxxxx.IBM.COM*, we would instead see *hostname.plexname.xxxxx.IBM.COM*. This was due to a DNS/WLM setup that we have for DB2. To get around this problem, we accessed a second TCP/IP stack that did not have the DNS/WLM function enabled. This is the short-term solution; the long-term solution is that the DNS/WLM function is slated to be removed from z/OS Communications Server in a future release of z/OS.

After we resolved the DNS issue, the LDAP server was able to obtain its Kerberos credentials. We verified this by seeing the following message in the JES SLAPDOUT log:

```
GLD0170I Kerberos authentication support has been enabled.
```

### Setting up the environment for debugging

To aid in debugging, we set the following environment variables in the `slapd.envvars` file. These variables pertain to the Generic Security Services Application Programming Interface (GSS API) Kerberos bind.

Table 12. GSS API environment variables we set in `slapd.envvars`

| Environment variable                             | Meaning                                      |
|--------------------------------------------------|----------------------------------------------|
| <code>_EUV_SVC_MSG_LOGGING=STDERR_LOGGING</code> | Directs logging to the standard error stream |
| <code>_EUV_SVC_DBG_MSG_LOGGING=1</code>          | Turns logging on                             |
| <code>_EUV_SVC_DBG=*8</code>                     | Defines the desired logging level            |

### Verifying Kerberos authentication support in the LDAP server

To verify that the LDAP server was indeed enabled for Kerberos authentication, we performed some LDAP searches.

**Steps to verify Kerberos authentication support in the LDAP server:** We performed the following steps to verify that our Kerberos authentication support was working properly:

1. We first needed to issue the **kinit** command from the z/OS UNIX shell to obtain a pass ticket, as follows:

```
kinit LDAP/hostname.xxxxx.ibm.com
```

In response to the password prompt, we entered the password that we defined earlier on the `ALTUSER` command.

2. We then issued the following **ldapsearch** command:

```
ldapsearch -h ip_address -V 3 -S GSSAPI -b "ou=Home Town, o=Your Company" "(objectclass=*)"
```

**Result:** The search returned the expected data.

Next, we tried some error conditions for further verification.

3. We removed the krb5ccname file from the \$HOME directory, which effectively negates the pass ticket we obtained in step 1. We then issued the **ldapsearch** command again and received the following error message:

```
ldap_sasl_bind_s: Unknown error
```

---

4. We reissued the **kinit** command from step 1 and the **ldapsearch** command from step 2 and the search again returned the expected data.
- 

5. We stopped the Kerberos server. We again issued the search command from step 2 and received the following error message:

```
ldap_sasl_bind_s: Unknown error
```

---

6. Finally, we restarted the Kerberos server and issued the search command again. The search once again returned the expected data.
- 

This concludes our verification of Kerberos authentication support in the LDAP server.

---

## Resolving a problem with malformed distinguished names in directory entries

The information in this section describes a problem we experienced whereby directory entries with a malformed distinguished name (DN) cause the UNLOAD utility and the **ldapsearch** command to end prematurely, before processing all of the data in a directory. When this error occurs, the LDAP server indicates that it is unable to read an entry from the database and returns the LDAP\_OPERATIONS\_ERROR code.

Upon investigating the problem, it appears that one or more of the entries in the directory were entered in such a way that the DN of the entry cannot be normalized by the z/OS LDAP Server. Exactly how such an entry occurred remains unclear. However, in this case, the specific problem appears to have been caused by an unescaped equal sign character (=) within an attribute value which was used as the relative DN (RDN) of the full DN of the directory entry.

Normally, when the LDAP server stores the DN value into the backing database (in this case, DB2), it should store an appropriately escaped character string so as to eliminate any ambiguity. However, in this case, the proper escaping was not performed. Later, when the LDAP server attempted to read and parse the entry from the database, the normalization of the entry's DN failed. The normalization failure caused the read operation to fail which, in turn, caused the search or unload operation to fail before returning all of the requested entries.

### Resolving the problem

If you suspect that you have encountered such a problem, follow your normal problem reporting procedure and contact IBM Level 2 service support for z/OS LDAP Server.

We worked directly with LDAP Server development to correct the problem in the DB2 data base. LDAP Level 2 service has also been made aware of this procedure.

## LDAP Server

The procedure is not for the faint of heart and requires the utmost caution to perform. Therefore, we have chosen *not* to publish the procedure here so that customers would only perform it under the direction and guidance of IBM service support personnel.

In addition, because we were not able to identify the root cause of the data corruption, LDAP development wants to hear from any customers who might experience this problem. Reporting the problem through the Level 2 service channel ensures that the appropriate development resources are available to help diagnose and resolve the root cause of the problem.

---

## Chapter 11. Setting up Security Server Network Authentication Service

Security Server Network Authentication Service for z/OS is the IBM z/OS program based on Kerberos Version 5. It provides Kerberos security services without requiring that you purchase or use a middleware product, such as Distributed Computing Environment (DCE). Such services include native Kerberos API functions, as well as the Generic Security Service Application Programming Interface (GSS-API) functions defined in Internet RFC 2078 and RFC 2744.

This chapter describes our experiences in setting up Network Authentication Service for z/OS, which we'll simply refer to as "Network Authentication Service" from here on.

---

### Experiences setting up Network Authentication Service

We set up and configured Network Authentication Service according to the information and instructions in the following documentation:

- *z/OS Program Directory*, GI10-0670
- *z/OS SecureWay Security Server Network Authentication Service Administration*, SC24-5926
- *z/OS SecureWay Security Server Network Authentication Service Programming*, SC24-5927
- *z/OS Security Server RACF Security Administrator's Guide*, SA22-7683

We found that setting up Network Authentication Service was not a smooth process, as we encountered several problems. Some of these problems may be unique to our test environment, while others are more generic in nature (which is a nice way of saying "we missed something"). Specifically, we encountered problems in the following areas:

- Missing HFS directories
- Failure to consult the program directory for initial customization
- Conflict with DCE

We'll describe these problems—and what we did to resolve them—in greater detail.

#### Missing HFS directories

We keep our own copies of the /etc and /var HFS directories and do not use the directory HFSs that our system build group provides. Therefore, the results of the z/OS installation process that creates additional subdirectories under /etc and /var were not available to us; we had to create the additional subdirectories ourselves.

##### Creating the missing subdirectories under /etc

During the installation process, the job EUVF.SEUVFSAM(EUVFMKDR) normally creates subdirectories under /etc. Accordingly, we manually created the following subdirectories under /etc and assigned them the appropriate permissions:

| Subdirectory       | Permissions |
|--------------------|-------------|
| /etc/skrb          | 755         |
| /etc/skrb/home     | 755         |
| /etc/skrb/home/kdc | 755         |

## Network Authentication Service

If you follow a similar practice of maintaining a separate /etc directory, be sure to create the additional subdirectories as above.

### Creating the missing subdirectories under /var

We were able to use the exec EUVF.SEUVFSAM(EUVFSMKV) to create the subdirectories under /var. We got lucky and found this job by looking in the program directory. We copied the exec to another dataset, modified it for our installation, and then ran it. We got the following output:

```
The EXEC to create the directories has begun.
It will run for a couple of minutes.
The EUVFMKDV EXEC ran at 13:00:19 on 9 Nov 2001
```

```
Created the following directories:
```

```
=====
/var/skrb 755
/var/skrb/creds 1777
```

```
Changed permission bits of existing directories:
```

```
=====
No permission bits were changed
```

```
Following directories already exist with proper permissions:
```

```
=====
No directories already existed with proper permission bits
```

```
Problems creating following directories:
```

```
=====
No problems while creating directories
```

```
Created the following symlinks:
```

```
=====
No symlink commands attempted
```

```
Deleted the following symlinks:
```

```
=====
No unlink commands attempted
```

```
Additional messages:
```

```
=====
No additional messages
```

```
End of EXEC.
The EUVFMKDV EXEC has completed with Return Code 0
READY
```

Again, if you follow a similar practice of maintaining a separate /var directory, you can use the EUVFMKDV exec to create the additional subdirectories as we did above.

## Failure to consult the program directory for initial customization

Be sure to consult the program directory for any initial customization steps. This is especially important when you have different people doing the z/OS installation and the security server setup. It's even more critical when you introduce other variations that are specific to your site, such as maintaining separate copies of HFS directories.

We did not consult the program directory at first; however, when we eventually did, we discovered an important step to use the `EUVF.SEUVFSAM(EUVFSMKV)` exec to create the subdirectories under `/var`, as we described earlier.

### Conflict with DCE

During our efforts to validate our Network Authentication Service customization, we encountered a conflict with DCE which, had we read *z/OS SecureWay Security Server Network Authentication Service Administration* more carefully, we could have avoided. The following steps describe how we discovered the conflict and how we resolved it.

### Steps to verify our Network Authentication Service customization

We performed the following steps to verify our Network Authentication Service:

1. We issued the following RACF command to add a Kerberos segment (KERB) to an existing user ID:

```
ALTUSER LDAPSRV PASSWORD(1dapsrv) NOEXPIRED KERB(KERBNAME(LDAP/xxx.xxx.xxx.ibm.com))
```

---

2. We issued the following RACF command to verify the KERB segment:

```
listuser 1dapsrv noracf kerb
```

#### Result:

```
USER=LDAPSRV

KERB INFORMATION

KERBNAME= LDAP/xxx.xxx.xxx.ibm.com
KEY VERSION= 001
KEY ENCRYPTION TYPE= DES DES3 DESD
```

---

3. We issued the following commands to change to the appropriate directory and add a keytab file:

```
cd /usr/lpp/skrb/bin
keytab add LDAP/xxx.xxx.xxx.ibm.com -p LDAPSRV -v 001
```

---

4. We issued the following command to verify the results of the **keytab** command:

```
keytab list
```

#### Result:

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

Principal: LDAP/xxx.xxx.xxx.ibm.com@xxx.xxx.IBM.COM
Key version: 1
Key type: 56-bit DES
Entry timestamp: 2001/10/17-09:01:48

Principal: LDAP/xxx.xxx.xxx.ibm.com@xxx.xxx.IBM.COM
Key version: 1
Key type: 56-bit DES using key derivation
Entry timestamp: 2001/10/17-09:01:48

Principal: LDAP/xxx.xxx.xxx.ibm.com@xxx.xxx.IBM.COM
```

## Network Authentication Service

```
Key version: 1
Key type: 168-bit DES using key derivation
Entry timestamp: 2001/10/17-09:01:48
```

---

5. We then issued the **kinit** command to validate the customization. Here is where we encountered the conflict with DCE as you can see by the command response.

```
kinit LDAP/xxx.xxx.xxx.ibm.com
```

**Result:**

```
EUVP08317A The DCE runtime library cannot communicate with the DCE kernel.
EUVS29107E Unable to parse principal name LDAP/xxx.xxx.xxx.ibm.com.
 Status code 0x141290df - No default realm specified in configuration file.
```

The references to DCE in the above command response alerted us to the conflict.

We found that in the chapter on configuring Network Authentication Service in *z/OS SecureWay Security Server Network Authentication Service Administration*, we had somehow missed the last bullet under the final step to make the program operational:

Update the users' UNIX System Services **.profile** by customizing the PATH environment variable to place **/usr/lpp/skrb/bin** ahead of any **/bin** or DCE subdirectory reference.

- 
6. We modified the PATH statement in our user's .profile file and then successfully issued the **kinit** command:

```
kinit LDAP/xxx.xxx.xxx.ibm.com
```

**Result:**

```
EUVF06017R Enter password:
```

---

This completes the verification of our customization. Network Authentication Service is now operational.

---

## Chapter 12. Using Enhanced ASCII functionality

This chapter describes some aspects of the Enhanced ASCII functionality that we experienced using the existing shell and utilities. In it, we focus on the behavior of Enhanced ASCII as influenced by various settings such as system parameters, environment variables, and command options. We do not focus on the programming aspects of Enhanced ASCII.

---

### Overview of Enhanced ASCII

z/OS is an EBCDIC platform. Enhanced ASCII functionality makes it easier to port internationalized applications developed on (or for) ASCII platforms to z/OS platforms by providing functions for EBCDIC-to-ASCII and ASCII-to-EBCDIC conversion. Enhanced ASCII does not change the basic EBCDIC nature of the z/OS system. For example, the z/OS shell and utilities continue to be EBCDIC programs. However, if you compile C programs as ASCII (using the new ASCII compiler option), the EBCDIC nature can be partially hidden.

Before Enhanced ASCII, there was no way for C programs to determine (or specify) the coded character set ID (CCSID) of the data in any given HFS file. Files encoded in a character set other than the standard EBCDIC-based IBM-1047 code page needed to be explicitly converted to EBCDIC (by using utilities such as **iconv**) before programs could perform I/O on them. Enhanced ASCII introduces file tagging and automatic conversion which, in many cases, eliminates the need to manually convert data.

Be sure to see *z/OS UNIX System Services Planning*, GA22-7800, for a complete description of the conditions under which you can use Enhanced ASCII and the limitations of its use.

### Understanding file tagging and automatic conversion

Enhanced ASCII functionality provides support for file tagging. File tags are a way to identify the code set of text data within HFS files.

A file tag is composed of a text flag (TXTFLAG) and a code set (CCSID):

- **TXTFLAG**: The text flag indicates whether or not a file contains uniformly encoded text data.
  - **TXTFLAG=ON** indicates that the file has uniformly encoded text data.
  - **TXTFLAG=OFF** indicates that the file has non-uniformly encoded text data.
- **CCSID**: The code set represents the coded character set in which the text data is encoded. The code set can be used for uniformly encoded text files or for files that contain mixed text and binary data.

Only files whose file tags have **TXTFLAG=ON** and a valid **CCSID** are eligible for automatic conversion.

When you enable Enhanced ASCII, z/OS UNIX needs to know whether a given text file is encoded as ASCII or EBCDIC. The file tag provides this information which can then be used to perform automatic conversion between the ASCII ISO8859-1 (CCSID=819) and EBCDIC IBM-1047 (CCSID=1047) code pages. If no file tag exists on a file, no conversion occurs during file I/O.

## Enhanced ASCII

You can also assign a CCSID to a program or thread in the shell. By default, the initial CCSID for every thread is 1047 (EBCDIC). There are several ways to change a program's CCSID to 819 (ASCII)—see *z/OS UNIX System Services Planning* for details.

Automatic conversion uses the CCSIDs of the programs and text files to determine the proper conversion method to use during I/O operations.

Figure 15 summarizes the concept of file tagging and automatic conversion.

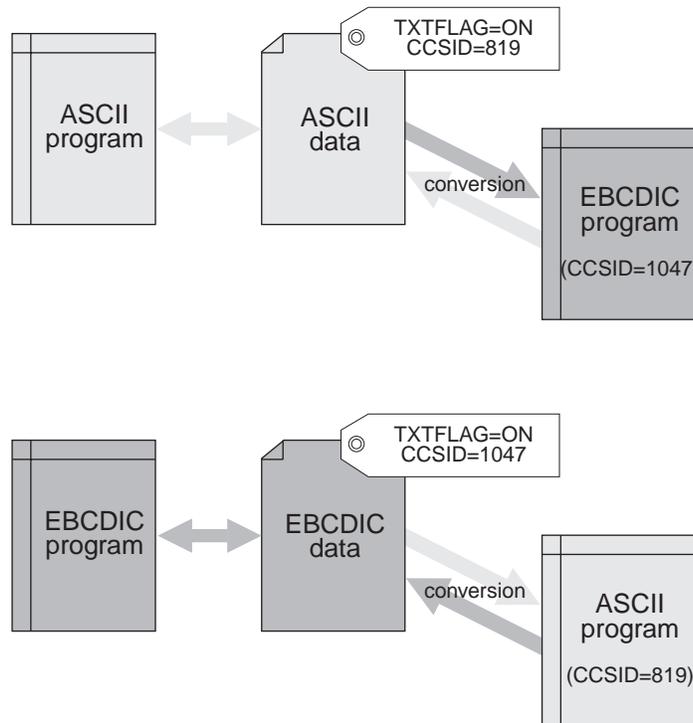


Figure 15. Conceptual summary of file tagging and automatic conversion

In the above figure, assume that automatic conversion is enabled. When an EBCDIC program (default CCSID=1047) performs I/O on a tagged file containing ASCII (CCSID=819) text (TXTFLAG=ON), the program sees EBCDIC data (automatically converted from ASCII) when it reads the file. On output, EBCDIC data from the program is automatically converted to ASCII before it is written to the file.

Similarly, when an ASCII program (CCSID=819) performs I/O on a tagged file containing EBCDIC (CCSID=1047) text (TXTFLAG=ON), the program sees ASCII data (automatically converted from EBCDIC) when it reads the file. On output, ASCII data from the program is automatically converted to EBCDIC before it is written to the file.

It is important to understand that file tagging and automatic conversion are independent operations. You can tag files without enabling automatic conversion, and vice versa. Setting a file tag does not force automatic code set conversion, but allows it to occur when automatic conversion is enabled. You can also mount a file system such that any untagged files or newly created files within the file system are

implicitly tagged for as long as the mount option is in effect. Therefore, it is possible to have many tagged files without any conversion occurring.

However, if you enable automatic conversion for the entire file system, every tagged file becomes subject to conversion by any program that reads or writes those tagged files. Thus, it is possible that a program might be processing converted data even though the program is not designed to support it.

**Example:** If automatic conversion is enabled, an EBCDIC program that reads a tagged ASCII file will be presented with EBCDIC data that has been automatically converted from the ASCII data in the file. However, if the program was designed to read the ASCII file as ASCII data, unpredictable results may occur.

For this reason, IBM recommends that you enable automatic conversion on the smallest scope necessary. “Controlling Enhanced ASCII functionality” on page 147 highlights several ways to do this.

## Implementing Enhanced ASCII

*z/OS UNIX System Services Planning* describes how to set up Enhanced ASCII and identifies some limitations on its use.

Since Enhanced ASCII is new in z/OS V1R2, there are no migration actions. In order to maintain compatibility with earlier releases, utilities will not tag files unless explicitly specified by the user.

**Coexistence considerations:** There is a coexistence consideration where Enhanced ASCII support is limited to z/OS UNIX files. It does not include MVS data sets, even if they can be accessed by z/OS UNIX. Conversion of read and write data is supported only by the standard z/OS UNIX read and write services (BPX1RED and BPX1WRT) in Assembler. In C, it is supported by read() and write(). Read and write for sockets is not supported; instead, separate TCP/IP services must be used.

Figure 16 illustrates how Enhanced ASCII support provides automatic conversion for a C program that performs I/O on an ASCII file.

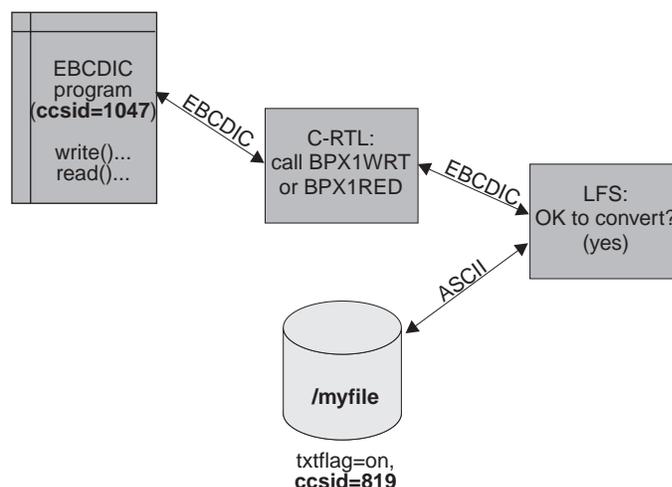


Figure 16. Automatic conversion for a C program performing I/O on an ASCII file

## Enhanced ASCII

**New command options for Enhanced ASCII:** Several shell commands have new options to handle file tags or automatic conversion. The commands that are affected are: **cp**, **df**, **find**, **iconv**, **ls**, **mount**, **mv**, **od**, **pax**, **tar**, **test** [...], and the **[[...]]** reserved-word command (reserved-word commands are described under **sh**). See *z/OS UNIX System Services Command Reference*, SA22-7802, for detailed information and syntax for these commands.

---

## Working with Enhanced ASCII

This section describes the methods to display and control the operation of Enhanced ASCII functionality, including how we set up our automount process to incorporate Enhanced ASCII support.

### Displaying file tag information

Several commands provide support for displaying file tag information for files or a file system. We include below some examples of commands we found to be useful. These examples only pertain to the display of file tag information. See the appropriate command references for complete descriptions and syntax for these commands.

The **chtag** command allows you to set, modify, remove, or display information in a file tag. The **-p** option displays the file tag.

**Example:** `chtag -p file_name`

**Result:**

```
t IBM-1047 T=on file_name
```

The **ls** command with the **-T** option also displays the file tag.

**Example:** `ls -T file_name`

**Result:**

```
t IBM-1047 T=on file_name
```

The **df** command with the **-v** option, the **DISPLAY OMVS,F** console command, and **ISHELL** all display the tag information for the file systems.

**Example:** `df -v mount_point`

**Result:**

```
Mounted on Filesystem Avail/Total Files Status
/mount_point (OMVS.TAG.FS) 448/2880 4294967150 Available
HFS, Read/Write
File System Owner : SYS1 Automove=Y Client=N
Filetag : T=on codeset=IS08859-1
```

**Example:** `display omvs,f`

**Result:**

```
HFS 1451 ACTIVE RDWR
NAME=OMVS.TAG.FS
PATH=/mount_point
OWNER=SYS1 AUTOMOVE=Y CLIENT=N
TAG=(TEXT, 819)
```

**Example:** From ISHELL, select **File\_systems** → **1. Mount table...**, then type action code **A** next to a file system.

**Result:**

Char Set ID/Text flag : 0819 0N

## Controlling Enhanced ASCII functionality

In this section we highlight some of the methods you can use to control the scope of Enhanced ASCII functionality.

There are three main areas of control:

1. You can use a combination of parmlib settings, environment variables, and run-time options to enable automatic conversion—either on a global scale or in a more localized application environment.
2. You can control the use of file tagging by explicitly creating file tags on individual files and by allowing (or disallowing) untagged files to take on an implicit or default file tag.
3. You can control the CCSID assigned to programs.

Figure 17 summarizes these three areas of control and the various commands, settings, and services that they involve.

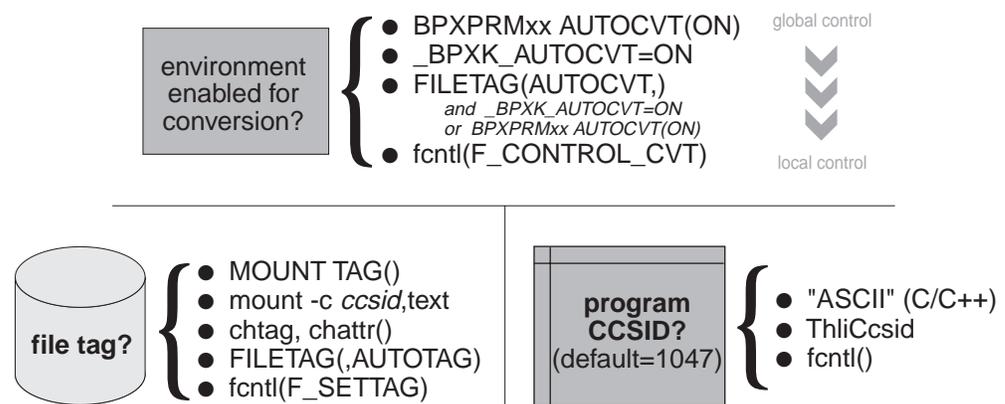


Figure 17. Summary of factors that control automatic conversion

The remaining sections in this chapter discuss the various commands, settings, and services that govern Enhanced ASCII functionality and, in some cases, our experiences with using them.

### Setting the AUTOCVT parameter in BPXPRMxx

The AUTOCVT(ON|OFF) parameter in the BPXPRMxx member of parmlib controls automatic conversion on a global scope. Individual programs can override AUTOCVT at a thread level. Therefore, AUTOCVT establishes a default control for programs that do not explicitly establish their own conversion options. The default setting is AUTOCVT(OFF).

Note that when AUTOCVT(ON) is in effect, every read or write operation for a file is checked to see if conversion is necessary. Thus, turning on AUTOCVT incurs a slight performance penalty even if no conversion occurs. For this reason, you might choose to leave AUTOCVT off to limit the scope of automatic conversion.

## Enhanced ASCII

You can also use the SETOMVS AUTOCVT=ON|OFF command to change the AUTOCVT setting; however, changing the conversion mode does not affect conversion of currently open files for which I/O has already begun. You can also override the global AUTOCVT setting by using the \_BPXK\_AUTOCVT environment variable, the FILETAG run-time option, or both.

Note that the **obrowse** and **oedit** commands only perform conversion with AUTOCVT(ON).

### C/C++ compiler and run-time support

Although we do not delve into specific programming aspects such as application design, development, or coding, we would like to point out the C/C++ compiler and run-time support for using Enhanced ASCII functionality.

Enhanced ASCII support provides the means to write z/OS C/C++ applications that execute with ASCII data representation. In order to use Enhanced ASCII support, a C or C++ module must specify the ASCII data representation at compile time. Specifically, application compile units must do the following in order to be bound to ASCII versions of external variables and interfaces at compile time:

- Use headers to declare external variables and interfaces that are used in the compile unit source code
- Specify the ASCII compile option

C programs that are compiled as ASCII use ASCII locales. These ASCII locales are produced by using the **-A** option of **localedef**.

New ASCII function versions and other support functions have been added to the z/OS C/C++ Run-Time Library to handle ASCII data in files, data manipulations, and conversions between ASCII and EBCDIC. The compile-time binding determines which ASCII or EBCDIC function versions are called during run-time execution.

In addition, existing C run-time library functions have been enhanced and new C functions have been introduced for setting, retrieving, and interpreting file tags. Existing functions that have been enhanced include: `fcntl()`, `stat()`, `fstat()`, `lstat()`, `fopen()`, `popen()`, and `freopen()`. New functions include: `__chattr()`, `__fchattr()`, `__CcsidType()`, `__CSNameType()`, `__toCcsid()`, and `toCSName()`. See the appropriate z/OS V1R2 C/C++ and Language Environment (LE) manuals for details.

### Using the Language Environment FILETAG() run-time option

The FILETAG(AUTOCVT|NOAUTOCVT,AUTOTAG|NOAUTOTAG) option in the C Run-Time Library (C-RTL) overrides the global AUTOCVT() setting for automatic conversion. It offers more granular control over automatic conversion for untagged files and standard streams that are opened as terminal files, and for control over whether certain open functions will tag new or empty files. The default is FILETAG(NOAUTOCVT,NOAUTOTAG).

**Note:** FILETAG is ignored under CICS.

You can also set the FILETAG run-time option with the \_CEE\_RUNOPTS environment variable or upon entry into the shell (for instance, **omvs runopts('FILETAG(...)').**

This run-time option is intended for use by the programmer whose application depends upon a certain setting and, thus, should not be overridden by the application user. This option should be set at compile time with a `#pragma runopts`, or at bind using a CEEUOPT CSECT that has previously been created.

**Note:** The FILETAG run-time option applies to the enclave. Nested enclaves do not inherit the setting of this run-time option; therefore, this option does not affect HFS files that are opened in the nested enclave.

**Using the FILETAG(AUTO CVT) run-time option:** The FILETAG(AUTO CVT) run-time option enables automatic conversion for untagged HFS files that are opened with `fopen()` or `freopen()` and sets up standard streams for conversion to EBCDIC if they refer to a terminal file (tty). With the FILETAG(AUTO CVT) option, automatic conversion only occurs if one of the following conditions is true:

- The environment variable `_BPXK_AUTO CVT` is set to "ON", or
- The environment variable `_BPXK_AUTO CVT` is unset and you define `AUTO CVT(ON)` in the active BPXPRMxx parmlib member

When you specify AUTO CVT as the first suboption of the FILETAG run-time option:

- Both the `fopen()` and `freopen()` functions enable automatic conversion for any untagged HFS files that they open. They also assign those files a CCSID of either 1047 or, if specified, the EBCDIC CCSID value from the `_BPXK_CC SID` environment variable.
- Automatic conversion to EBCDIC occurs when writing to standard streams that refer to a tty as if they were tagged as IBM-1047. This is helpful because data written by ASCII compiled programs is usually encoded as ISO8859-1 and needs to be converted before outputting to the terminal.

The enablement of automatic text conversion for the standard streams only occurs when the application is invoked with `exec()` (for instance, when the UNIX shell gives control to a program entered on the command line) and the file descriptors for the standard streams are already open, untagged, and associated with a tty.

**Using the FILETAG(,AUTOTAG) run-time option:** The FILETAG(,AUTOTAG) run-time option activates automatic tagging of new or empty HFS files that are opened with `fopen()`, `freopen()`, or `popen()`.

When you specify AUTOTAG as the second suboption of the FILETAG run-time option:

- The `fopen()` and `popen()` functions with the text option automatically tag new or empty files upon first write. Programs that open files in this manner are already set up for tagging and require the least effort to set up automatic conversion.
- The `fopen()` and `freopen()` functions allows for the tagging of new or empty HFS files. If the open mode is not binary then, upon first write, the open file is tagged as text (`TEXTFLAG=ON`) along with the writer's Program CCSID. If the open mode is binary then, upon first write, the open file is tagged as non-text (`TEXTFLAG=OFF`) along with the writer's Program CCSID.

**Note:** Automatic tagging only occurs upon first write and only if the file has been opened with `fopen()` or `freopen()`.

- The pipe that the `popen()` function opens for communication between the parent and child process is tagged with the writer's Program CCSID upon first write. For instance, the call `popen(some_command, "r")` returns a stream that is tagged in the child process with the child process' Program CCSID upon first write.

**Examples:** The following example scenarios further illustrate the use of the FILETAG(,AUTOTAG) run-time option:

1. Use **touch** to create a new, empty file.

## Enhanced ASCII

Because **touch** uses `open()`, AUTOTAG has no effect on the new file. AUTOTAG only affects files that are opened with `fopen()` or `freopen()`.

2. Use **touch** to create a new, empty file, put data in it, then empty it.

As in the previous example, because **touch** opens the file, AUTOTAG does not tag the file even when data is written to it.

3. Use **ed** to create a new, empty file, put data in it, then empty it.

In this case, automatic tagging occurs upon the first write to the file. Subsequently emptying the file does not change the tag.

### Using the TSO MOUNT command with the TAG() parameter

The TAG() parameter of the TSO MOUNT command specifies whether or not to implicitly set file tags for untagged files in the mounted file system. Implicit, in this case, means that the tag is not permanently stored with the file. Rather, the tag is associated with the file during reading or writing, or when `stat()` type functions are issued. When the file system is unmounted, all implicit tags are removed, unless tagged in another manner (such as with the **chtag** command).

To use the TAG() parameter, you must specify TEXT or NOTEXT, along with a CCSID, as follows:

```
TAG(TEXT|NOTEXT,ccsid)
```

where:

**TEXT** Specifies that each untagged file is implicitly marked as containing pure text data that can be converted.

**NOTEXT** Specifies that none of the untagged files in the file system are automatically converted during file reading and writing.

*ccsid* Identifies the coded character set identifier to be implicitly set for the untagged files. The *ccsid* must be a decimal value from 0 to 65535. Other than this, the value is not checked to see if it represents a valid code page nor whether the corresponding code page is installed.

**Example:** The following is an example of the TSO MOUNT command with a TAG() parameter that indicates that untagged files in the file system are to be implicitly tagged as containing ASCII text:

```
mount filesystem('OMVS.TAG.FS') type(hfs) mode(rdwr) mountpoint('/tag') TAG(TEXT,819)
```

The following are some other examples of the TAG() parameter:

- TAG(TEXT,819) identifies text files containing ASCII data that is eligible for conversion
- TAG(TEXT,1047) identifies text files containing EBCDIC data that is eligible for conversion
- TAG(NOTEXT,65535) identifies files as containing binary or unknown data
- TAG(NOTEXT,0) is equivalent to not specifying the TAG parameter at all

### Using the mount shell command with the -c option

The -c option of the **mount** shell command specifies whether or not to implicitly set file tags for untagged files in the mounted file system. The concept is identical to the TSO MOUNT command with the TAG() parameter described earlier.

To use the -c option, you must specify a *ccsid* along with either `text` or `notext`, as follows:

```
-c ccsid,text|notext
```

The meaning of *ccsid*, *text*, and *notext* is the same as for the TAG() parameter of the TSO MOUNT command.

#### A heads-up about syntax

The order of the arguments on the `-c` option of the **mount** shell command are the opposite of those in the TAG() parameter of the TSO MOUNT command. The arguments are positional and the *ccsid* must precede *text* or *notext*, as shown above.

The format of the **mount** shell command in the version of *z/OS UNIX System Services Command Reference* that we looked at, as well as the man pages, shows the *ccsid* and *text|notext* in the wrong order. This will be corrected in the next documentation update. The response to the `mount ?` command does show the correct format.

**Example:** The following is an example of the **mount** command with a `-c` option that indicates that untagged files in the file system are to be implicitly tagged as containing ASCII text:

```
mount -f OMVS.TAG.FS -t HFS -c 819,text /tag
```

### Using the mount panel under TSO ISHELL

ISHELL invokes the z/OS ISPF shell—a panel interface that helps you set up and manage z/OS UNIX System Services functions. You can use the ISHELL panel interface to mount filesystems with tag information. The concept is the same as the TSO MOUNT command with the TAG() parameter described earlier.

The two fields on the ISHELL mount panel that relate to file tagging are:

|                                |                                                                                                |
|--------------------------------|------------------------------------------------------------------------------------------------|
| <b>Character set ID</b>        | Specify the CCSID in this field                                                                |
| <b>Text conversion enabled</b> | Select this option to specify the equivalent of TEXT or leave it unselected to specify NOTEXT. |

### Setting up automount to tag files

You can specify file tag information for automounted file systems. To do this, there is a new **tag** configuration parameter for automount. The syntax of the **tag** parameter is similar to the TAG() parameter of the TSO MOUNT command described earlier:

```
tag text|notext,ccsid
```

**Steps for setting up automount to tag files:** The following procedure describes how we set up our automount process to incorporate the use of file tags.

1. First, we created a MapName file in `/etc`. For this example, we'll call it `/etc/tag.map` and it contains the following:

```
name AUTOMTAG
type HFS
filesystem OMVS.USER.AUTOMTAG.FS
mode rdwr
duration 00
delay 00
tag text,1047
```

The last line specifies the new tag parameter. In this case, we're specifying EBCDIC text files.

## Enhanced ASCII

2. We then add to the `/etc/auto.master` file (which is our automount configuration file) the directory whose mounting we want automount to control with the tag parameter. For this example, we'll use the `/tag` directory. The following shows the contents of our `/etc/auto.master` file:

```
/u /etc/u.map
/tag /etc/tag.map
```

---

3. Now we issue the automount command with the `-s` option to check the syntax of the configuration file. (No directories are automounted yet.)

```
/usr/sbin/automount -s /etc/auto.master
```

---

4. We then make sure that the directory that is being monitored (`/tag`) has been created with the proper permissions.
- 

5. We ensure that the file system data set (OMVS.USER.AUTOMTAG.FS) is allocated and cataloged.
- 

6. Now, we activate automount with our new configuration:

```
/usr/sbin/automount /etc/auto.master
```

---

7. We then access the `/tag/AUTOMTAG` directory to cause it to be automounted:

```
cd /tag/AUTOMTAG
```

---

This completes our automount setup. To check that it worked, we can display the file system information and check the file tag parameters, as follows:

**Example:** `df -v /etc/tag/AUTOMTAG`

### Result:

```
Mounted on Filesystem Avail/Total Files Stat
/tag/AUTOMTAG (OMVS.USER.AUTOMTAG.FS) 784/960 4294967294 Available
HFS, Read/Write
File System Owner : SYS2 Automove=Y Client=N
Filetag : T=on codeset=IBM-1047
```

## Using environment variables

The following are some new environment variables that you can use to control automatic conversion.

### `_BPXK_AUTOCVT`

Controls the conversion of data between ASCII and EBCDIC code sets. It can be set ON or OFF to allow users to globally activate or deactivate automatic conversion of tagged HFS files. This variable behaves much like the `AUTOCVT()` parameter in `BPXPRMxx` and will override that parameter's setting.

### `_BPXK_CCSIDS`

Defines a corresponding pair of valid EBCDIC and

ASCII CCSIDs that are to be used for automatic conversion or for tagging new files.

Note the following about these environment variables:

- In the `/bin/tcsh` shell, use the **setenv** command to set these environment variables.
- In the `/bin/sh` shell, use the **export** command to set these environment variables.
- You can also set these variables upon entry into the shell. For instance, **omvs runopts('ENVAR(\_BPXK\_AUTOCVT=ON)')**. Envars that are defined via run-time options are set upon initialization of the parent program.
- If you use built-in commands and you set or change the setting of these variables while in the `/bin/sh` shell, you must reinitialize the shell environment in order for the built-in commands to recognize the new settings. An example of this is the **cat** command.

### Using redirection in `/bin/sh` and `/bin/tcsh` shell variables

The following new shell variables control the conversion of untagged files:

`_TAG_REDIR_IN`, `_TAG_REDIR_OUT`, and `_TAG_REDIR_ERR`. Possible settings for these variables are:

|                                 |                                                                                                                                                                                  |
|---------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>_TAG_REDIR_IN=TXT</code>  | Redirected stdin overrides the file's text flag (TXTFLAG) and treats it as if it were tagged as: TXTFLAG=ON, CCSID=existing file tag CCSID. This has no effect if the CCSID=0.   |
| <code>_TAG_REDIR_IN=BIN</code>  | Redirected stdin overrides the file's TXTFLAG and treats it as if it were tagged as: TXTFLAG=OFF, CCSID=existing file tag CCSID. This effectively disables automatic conversion. |
| <code>_TAG_REDIR_OUT=TXT</code> | Redirected stdout will be tagged as: TXTFLAG=ON, CCSID=program CCSID at the time of the first write (if not already tagged).                                                     |
| <code>_TAG_REDIR_OUT=BIN</code> | Redirected stdout will be tagged as: TXTFLAG=OFF, CCSID=program CCSID at the time of the first write (if not already tagged).                                                    |
| <code>_TAG_REDIR_ERR=TXT</code> | Redirected stderr will be tagged as: TXTFLAG=ON, CCSID=program CCSID at the time of the first write (if not already tagged).                                                     |
| <code>_TAG_REDIR_ERR=BIN</code> | Redirected stderr will be tagged as: TXTFLAG=OFF, CCSID=program CCSID at the time of the first write (if not already tagged).                                                    |

**Using the z/OS UNIX shell, `/bin/sh`:** The automatic conversion shell variable can be specified for one command or for multiple commands within a shell session or shell script. If the variable is exported, it will affect child shells—that is, nested shell scripts. Note that because the standard shell execution performs redirection before variable assignment, the syntax for specifying the shell variable for one command is, for example:

```
(_TAG_REDIR_OUT=TXT; command >file)
```

These variables can also be used in pipelined commands to tag the stdout of each command that writes to a pipeline, or the stdin of each command that reads from a pipeline.

## Enhanced ASCII

**Using the C shell, /bin/tcsh:** The automatic conversion shell variable can be specified for one command or for multiple commands within a tcsh shell session or shell script. If the variable is set in a user's .tcshrc file, then it will affect child shells—that is, nested shell scripts. Note that because the standard tcsh shell execution performs redirection before variable assignment, the syntax for specifying the shell variable for one command is, for example:

```
(set _TAG_REDIR_OUT=TXT; command >file)
```

These variables can also be used in pipelined commands to tag the stdout of each command that writes to a pipeline, or the stdin of each command that reads from a pipeline.

Note the following about the above shell variables:

- In the /bin/tcsh shell, use the **set** command to set these shell variables.
- In the /bin/sh shell, you can use the **export** command to set these shell variables.
- You can also set these variables upon entry into the /bin/sh shell only. For instance, **omvs runopts('ENVAR(\_TAG\_REDIR\_OUT=TXT)')**.
- If you use built-in commands and you set or change the setting of these variables while in the /bin/sh shell, you must reinitialize the shell environment in order for the built-in commands to recognize the new settings. An example of this is the **cat** command.

### Using the chtag command

The **ctag** command allows you to set, modify, remove, or display information in a file tag. You must have write permission on the file or be a superuser in order to use the **ctag** command.

The following are some examples of using the ctag command. For more information including complete syntax, see *z/OS UNIX System Services Command Reference*.

**Example:** To specify a text file with IBM-1047 code set, issue:

```
ctag -tc IBM-1047 filename
```

**Example:** To specify a binary file, issue:

```
ctag -b filename
```

**Example:** To specify a file of mixed binary and text data with a new code set of ISO8859-1, issue:

```
ctag -mc ISO8858-1 filename
```

**Example:** To remove a file tag from a file, issue:

```
ctag -r filename
```

**Example:** To display the file tag information for a file, issue:

```
ctag -p filename
```

**Result:** Sample output for this command might look something like this:

```
t IBM-1047 T=on file1
- untagged T=on file2
b binary T=off file3
m ISO-8859-1 T=off file4
- untagged T=off file5
b binary T=on file6
```

where the characters in the first column mean the following:

- t = text
- b = binary
- m = mixed
- = untagged



---

## Chapter 13. Using PKI Services

This chapter describes how we installed, configured, and tested PKI Services in our environment.

---

### Introduction to PKI Services

PKI Services is a new Security Server component that is available starting in z/OS V1R3. This service provides a certificate authority (CA) for the z/OS environment and enables you to issue and administer digital certificates rather than purchase them from an external certificate authority. PKI Services supports Public Key Infrastructure for X.509 version 3 (PKIX) and Common Data Security Architecture (CDSA) cryptographic standards.

We used *z/OS Security Server PKI Services Guide and Reference*, SA22-7693, to install and configure PKI Services in our environment and we found the document to be an invaluable reference. The setup instructions were well done and led us step-by-step through the process. They also provide several decision tables and other reference tables to help you make configuration decisions and allow you to fill in the information along the way for your particular environment. We found this to be very helpful.

Despite having great documentation though, we did encounter a few difficulties along the way. We'll point out the problems we had and what we did to resolve them as we describe our activities in the following sections.

---

### Installing and configuring prerequisite software

The chapter on installing and configuring prerequisite products in *z/OS Security Server PKI Services Guide and Reference* identifies the following three mandatory z/OS elements and components, and one optional component:

- IBM HTTP Server — see our experiences in “Installing and configuring IBM HTTP Server”
- OCSF and OCEP — see our experiences in “Installing and configuring OCSF and OCEP” on page 158
- LDAP Server— see our experiences in “Installing and configuring LDAP Server” on page 159
- (Optional) Integrated Cryptographic Service Facility (ICSF) — we did not use ICSF

### Installing and configuring IBM HTTP Server

We already have IBM HTTP Server installed and configured in our environment. We did not have to make any changes to our configuration at this time. Following the PKI Services documentation, we made note of the following two pieces of information for future use:

*Table 13. Information about our IBM HTTP Server configuration*

| IBM HTTP Server information                | Values we used              |
|--------------------------------------------|-----------------------------|
| IHS fully qualified domain name            | xxx.ibm.com                 |
| Full UNIX path name of our httpd.conf file | /Z0/dgw_pet/httpdz0pki.conf |

### Installing and configuring OCSF and OCEP

Open Cryptographic Services Facility (OCSF) is a component of the Cryptographic Services base element of z/OS; OCSF Security Level 3 is an optional feature of z/OS and works with OCSF to provide stronger encryption (greater than 64 bits). Open Cryptographic Enhanced Plug-ins (OCEP) is a component of the Security Server optional feature of z/OS.

#### Configuring OCSF

We already had OCSF and the OCSF Security Level 3 feature installed (see below for information on how to determine if this is installed) as part of our normal z/OS installation. We needed to configure OCSF on our system Z0, which is the image we'll be using for PKI Services. To do this, the PKI Services documentation referred us to the chapter on configuring and getting started in *z/OS Open Cryptographic Services Facility Application Programming*, SC24-5899.

**Steps for configuring OCSF:** We followed the instructions in the OCSF documentation; however, some previous work had been done to set up OCSF on other images in our sysplex, so we were able to skip the parts that were already done.

1. Set up the necessary security authorizations.

Because we already had OCSF in our environment, we did not have to define any of the OCSF data sets to RACF. However, we did ensure that the appropriate files were APF-authorized.

---

2. Run the installation script.

The OCSF documentation states that you must reinstall and run the configuration scripts with every new release of z/OS. Because we originally installed OCSF under a prior release of z/OS, we reran the installation script now. We did the following:

- a. We created a user ID of OCSFADM and gave it READ access to the BPX.SERVER facility class and to the following data sets:
  - CDS.CSSM
  - CDS.CSSM.CRYPTO
  - CDS.CSSM.DATALIB

**Note:** Later on in the setup of PKI Services, you will run an exec called IKYSETUP that creates the daemon and user identities that it requires and will give it the appropriate access. The default user ID assigned to the PKI Services daemon is PKISRVD. We found that we also had to give this user ID READ access to the same three data sets (above) as we did for the OCSFADM user ID. We'll mention this again later, after the step to run IKYSETUP.

- b. We logged on to the OCSFADM user ID and successfully ran the ocsf\_install\_crypto installation script from the /usr/lpp/ocsf/bin directory.
- 

3. Run the installation verification procedure (IVP).

We successfully ran the ocsf\_baseivp script from the /usr/lpp/ocsf/ivp directory.

---

This completes the configuration of OCSF.

**Determining if the OCSF Security Level 3 feature is installed:** If this feature is installed, the files `/usr/lpp/ocsf/lib/cssmmanp.dll` and `/usr/lpp/ocsf/lib/cssmusep.dll` should point (as links) to `cssmmanp_sl3.dll` and `cssmusep_sl3.dll`, respectively. If just the OCSF base is installed, these files will point to `cssmmanp_sl2.dll` and `cssmusep_sl2.dll`, respectively.

### Configuring OCEP

To configure OCEP, the PKI Services documentation referred us to the chapter on configuring and getting started in *z/OS SecureWay Security Server Open Cryptographic Enhanced Plug-ins Application Programming*, SC24-5925.

**Steps for configuring OCEP:** We followed the instructions in the OCEP documentation.

**Note:** We are using the default value, `/var/ocsf`, for the location of the OCSF registry.

1. Verify the OCSF installation and configuration.

Because we had just finished the OCSF configuration, we did not need to verify that it was complete and working properly.

---

2. Configure the OCEP installation.

- a. During the setup of PKI Services, you will be asked to run an exec called IKYSETUP which will create the daemon and user identities that it requires and will give it the appropriate access. Therefore, we did not have to do anything to authorize the daemon and user identities, nor to grant access to RACF facility class profiles.
  - b. As for establishing program control, LE already runs with program control in our environment and we are not using ICSF, so we did not have to do anything else.
  - c. We did verify that the `/usr/lpp/ocsf/addins/ibmoceptp.so` file was APF-authorized.
- 

3. Install the OCEP code.

- a. We did not rerun the OCSF IVP (`ocsf_baseivp`) because we had just done so earlier.
  - b. We successfully ran the `ocep_install` script from the `/usr/lpp/ocsf/bin` directory.
- 

4. Verify the OCEP installation.

We successfully ran the `ocep_ivp` from the `/usr/lpp/ocsf/ivp` directory.

---

This completes the configuration of OCEP.

## Installing and configuring LDAP Server

In our environment, LDAP Server for the TDBM DB2 backend was already installed and configured, so we did not have to do any setup work.

## PKI Services

We recorded the following LDAP configuration information for use later on:

Table 14. Information about our LDAP configuration

| LDAP information                          | Values we used                          |
|-------------------------------------------|-----------------------------------------|
| Administrator's distinguished name        | "cn=LDAP Administrator, o=Your Company" |
| Administrator password                    | <i>password</i>                         |
| LDAP fully qualified domain name and port | yyy.ibm.com:389                         |
| Suffix                                    | "o=SAM Company, c=US"                   |

## Configuring the system for PKI Services

After we completed the setup of the prerequisite software, we continued with the instructions in *z/OS Security Server PKI Services Guide and Reference* to configure our system for PKI Services. In this section, we describe what we did for each of the following tasks:

- "Running IKYSETUP to perform RACF administration"
- "Configuring the UNIX runtime environment" on page 162
- "Tailoring the LDAP configuration for PKI Services" on page 163
- "Updating the IBM HTTP Server configuration and starting the server" on page 164
- "Tailoring the PKI Services configuration file for LDAP" on page 166
- "Creating VSAM data sets and starting PKI Services" on page 166

## Running IKYSETUP to perform RACF administration

Much of the work for this task actually falls under the "Before you begin" section in the PKI Services documentation, where you need to decide on the values of the variables that IKYSETUP uses. To help you plan these values, the documentation provides several tables that list the variables, their descriptions, default values, and space for you to record the values you decide to use. We went through the tables and filled in the requested information.

This was our first attempt at running the IKYSETUP exec, so we decided to change only the variables whose values had to change; we accepted the default values for the rest.

The following are the variables for which we supplied new values:

Table 15. Values we specified for IKYSETUP variables

| Variable name                            | Values we used                                                                                              |
|------------------------------------------|-------------------------------------------------------------------------------------------------------------|
| ca_dn<br>(See note 1 on page 161 below.) | "ou=PKI Certificate Authority"<br>"o=SAM Company"<br>"c=US"                                                 |
| pkigroup_mem.0                           | 1                                                                                                           |
| pkigroup_mem.1                           | "WEBADM"                                                                                                    |
| web_dn                                   | ,<br>"CN('xxx.ibm.com')",<br>"O('Your Company')",<br>"L('Poughkeepsie')",<br>"SP('New York')",<br>"C('US')" |

Table 15. Values we specified for IKYSETUP variables (continued)

| Variable name                   | Values we used |
|---------------------------------|----------------|
| webservr<br>(See note 2 below.) | WEBADM         |

**Notes:**

1. The documentation says that the suffix of the PKI Services CA distinguished name (DN) must match the LDAP suffix. We had originally defined our LDAP suffix to be "o=SAM Company", so we assumed that we also had to set the o= value for the ca\_dn variable to "SAM Company". When we coded the ca\_dn value as shown above but had the LDAP suffix set to "o=SAM Company", we got the following messages when we tried to start the PKI Services daemon, PKISERVD (actual step occurs later on):

```
IKYC009I LDAP post unsuccessful for object id = 100, state =0x2150000, status =6032:
No description found for this error code
IKYP008E Directory post unsuccessful. LDAP data library module rc=6032
IKYC008I Error 5125 creating an CSSM_DL_DB_RECORD_CRL entry for CRL to LDAP for OU=PKI
Certificate Authority,O=Your Company,C=US: No description found for this error code
```

The problem was that, although our assumption was true, it also means that the ca\_dn hierarchy cannot have anything coded past the o= value. We resolved this by changing the LDAP suffix to "o=SAM Company, c=US" (as shown earlier in Table 14 on page 160).

2. In the PKI Services documentation, the section "IKYSETUP variables you can optionally change" lists the *webservr* variable with a default value of WEBSRV. Because we run our HTTP server under a user ID of WEBADM, we had to change the owner of the certificate created by IKYSETUP from WEBSRV to WEBADM. We originally missed this and when we tried to start the second instance of the HTTP server (actual step occurs later on), we got the following messages:

```
IMW6304E Key Data Base not read. Please check keyfile existence, permission and
ownership. Using Default root keys.
IMW6310E SSL support initialization failed, server will run only in non-secure mode
without listening on ssl port
IMW0501E Accept soft error: Errno: 121 Errno2: 12b80289 Error: EDC5121I Invalid
argument. (errno2=0x12B80289)
IMW0502E Accept () soft error count excessive...shutting down
```

Once we changed the certificate owner to WEBADM, we were able to successfully start the HTTP server instance.

**Steps for running IKYSETUP**

**Before we began:** We decided upon the values to specify for the IKYSETUP variables that we wanted to change, as we described earlier.

**Note:** You must run IKYSETUP from a user ID that has the RACF SPECIAL attribute. (We didn't realize this at first, as you'll see below.)

We performed the following steps:

1. We copied the IKYSETUP member from SYS1.SAMPLIB to our *hlq*.JOBS data set.
2. We edited the IKYSETUP exec and made the changes to the variable values, as we described earlier, and saved the changes.

3. From our OCSFADM user ID, we ran IKYSETUP with the RUN(NO) parameter.

---
4. We reviewed the log data set and the information looked fine.

---
5. We ran IKYSETUP with the RUN(YES) parameter.  
We ran into a problem because our OCSFADM user ID was not authorized to issue some of the RACF commands that IKYSETUP was attempting.  
We re-ran IKYSETUP with RUN(YES) using our SETUP user ID (which has the RACF SPECIAL attribute). This time we got a few return codes of 4, but they didn't seem to indicate any major problems, and the commands that had failed the first time around now ran successfully.

---

This completes the use of IKYSETUP.

As we mentioned earlier in "Configuring OCSF" on page 158, IKYSETUP creates the PKISRVD user ID. However, at this point, the PKISRVD user ID does not have the proper authority to access the OCSF data sets. If you attempt to start the PKISRVD daemon (which actually occurs later on), you will get the following error messages:

```
ICH408I USER(PKISRVD) GROUP(SYS1) NAME(PKI SRVS DAEMON) CDS.CSSM CL(FACILITY)
INSUFFICIENT ACCESS AUTHORITY
ACCESS INTENT(READ) ACCESS ALLOWED(NONE)
IKYP016I THE PKI SERVICES RUNTIME ENVIRONMENT COULD NOT BE INITIALIZED
ICH408I USER(PKISRVD) GROUP(SYS1) NAME(PKI SRVS DAEMON) CDS.CSSM CL(FACILITY)
INSUFFICIENT ACCESS AUTHORITY
ACCESS INTENT(READ) ACCESS ALLOWED(NONE)
INSUFFICIENT ACCESS AUTHORITY
ACCESS INTENT(READ) ACCESS ALLOWED(NONE)
```

We took the next step to resolve this problem:

6. We issued the following RACF commands to properly authorize the PKISRVD user ID:  
PERMIT CDS.CSSM CLASS(FACILITY) ID(PKISRVD) ACC(READ)  
PERMIT CDS.CSSM.CRYPTO CLASS(FACILITY) ID(PKISRVD) ACC(READ)  
PERMIT CDS.CSSM.DATALIB CLASS(FACILITY) ID(PKISRVD) ACC(READ)  
SETROPTS RACLIST(FACILITY) REFRESH
- 

This completes the IKYSETUP and RACF administration task.

## Configuring the UNIX runtime environment

We reviewed the chapter on configuring the UNIX runtime environment in *z/OS Security Server PKI Services Guide and Reference* and we decided to accept the default values for all of the environment variables and the contents of the `pkiserv.conf` configuration file.

We decided to copy the `pkiserv.envars` file into our `/etc/pkiserv` runtime directory, just in case we need to update it later. Accordingly, we also updated the PKISRVD startup proc to point to this new location.

We did not perform any of the optional steps described in the documentation.

## Tailoring the LDAP configuration for PKI Services

We followed the steps to tailor the LDAP configuration for PKI Services by loading the user schema file so that the LDAP server understands the format of the entries that will be stored in the directory. In doing so, we discovered a conflict with the LDAP server setup work that we did prior to z/OS V1R3, but we resolved the problem, as we describe below.

### Steps for tailoring the LDAP configuration for PKI Services

We performed the following steps:

1. We copied the `/usr/lpp/ldap/etc/schema.user.ldif` file to our `/tmp/pki` directory.

2. We edited the `schema.user.ldif` file and updated the “dn:” line to add the same suffix that we defined earlier:

```
dn: cn=schema, o=SAM Company, c=US
```

3. We attempted to load the schema file. Since our LDAP server runs on a different z/OS system than the one on which we’re setting up PKI Services, we had to specify the `-h` parameter on the `ldapmodify` command. We issued the following command from the `/tmp/pki` directory:

```
ldapmodify -h yyy.ibm.com -D "cn=LDAP Administrator,o=Your Company"
-w secret -V 3 -f schema.user.ldif
```

**Result:** We received the following messages:

```
ldap_modify: Type or value exists
ldap_modify: additional info: R004086 Entry CN=SCHEMA already contains attribute
attributetypes, value=(1.3.18.0.2.4.708 NAME abstract' DESC 'Provides an abstract
of a document entry.' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 USAGE userApplications)
(tdbm_modify.c|1.80.1.2|1293)
```

We assumed that this was an informational message and that the `ldapmodify` command would still execute, so we moved on to the next task. However, we later learned that the above messages indicate an error and that the `ldapmodify` command did not complete. We got to the step where we tried to start the PKISERVD daemon and we got the following messages:

```
IKYC009I LDAP post unsuccessful for object id = 100, state = 0x2150000, status = 6065:
No description found for this error code
IKYP008E Directory post unsuccessful. LDAP data library module rc = 6065
```

We referenced these messages in *z/OS Open Cryptographic Services Facility Application Programming* and found that the error associated with return code 6065 is `LDAPDL_OBJECT_CLASS_VIOLATION`. It was then that we realized that the schema had not loaded successfully.

We had set up LDAP Server in our environment prior to z/OS V1R3. During that initial setup, we loaded the `schema.user.ldif` file. As a result, when we now tried to load the entire `schema.user.ldif` file for PKI Services, the load failed. APAR OW53820 addresses this problem.

To get around this problem, instead of loading the entire `schema.user.ldif` file, we had to load three individual files:

- `EntrustPKIV4.ldif`
- `EntrustPKIV5.ldif`

## PKI Services

- RFC2587.ldif

These files are located along with the schema.user.ldif file in the /usr/lpp/ldap/etc directory. We basically had to repeat the above three steps for each of the three individual files. The remaining steps show what we did:

4. We copied the three individual files from /usr/lpp/ldap/etc to our /tmp/pki directory.

- 
5. We edited each of the files and updated the "dn:" line to add our suffix, as we did in step 2 on page 163.

- 
6. We issued the following commands from the /tmp/pki directory to load the schema:

```
ldapmodify -h yyy.ibm.com -D "cn=LDAP Administrator,o=Your Company"
-w secret -V 3 -f EntrustPKIV4.ldif
ldapmodify -h yyy.ibm.com -D "cn=LDAP Administrator,o=Your Company"
-w secret -V 3 -f EntrustPKIV5.ldif
ldapmodify -h yyy.ibm.com -D "cn=LDAP Administrator,o=Your Company"
-w secret -V 3 -f RFC2587.ldif
```

**Result:** Each of the above three commands displayed the following response: modifying entry cn=schema, o=Your Company.

---

This completes the task of tailoring the LDAP configuration for PKI Services.

## Updating the IBM HTTP Server configuration and starting the server

PKI Services requires running two HTTP server instances, each with its own separate configuration file. The tasks in this section involve updating the IBM HTTP Server configuration files for the two server instances and starting the server instances.

### Updating the IBM HTTP Server configuration files

We followed the instructions in the PKI Services documentation to update the HTTP server configuration files, with the following minor variation:

The documentation suggests making a copy of the default httpd.conf file, then updating the original file for the first server instance and updating the copy for the second server instance. However, for our test environment, we decided to leave our existing configuration file, httpdz0.conf, intact and instead made two copies of it: httpdz0pki.conf and httpd1443.conf. We used the httpdz0pki.conf file for the first server instance and httpd1443.conf for the second server instance.

We followed all of the remaining instructions to update both configuration files. Since we copied our second configuration file, httpd1443.conf, from our original configuration file, we deleted all of the existing protection, protect redirect, pass and exec directives, as indicated in the instructions. Because we deleted all of those existing directives, when we got to a later step we didn't need to worry about putting the new directives after any of the existing ones.

**Note:** We also needed to comment out the FastCGI directives in the second configuration file. Because we copied these files from our original

configuration file, they both contained identical FastCGI directives with the same port number. We realized this conflict when we tried to start the second server instance (below).

### Starting the HTTP server instances

In our environment, we specify ICSPARM in our HTTP server startup proc to define which configuration file to use. In order to start these server instances with different configuration files, we commented out the ICSPARM line in the startup proc. We then specified the ICSPARM (with a specific configuration file name) on the startup command for each server instance (see step 2 below).

**Steps for starting the HTTP server instances:** We did the following to start the two HTTP server instances:

1. We verified that LDAP Server was running.

2. We issued the following MVS command to start the first HTTP server instance:

```
S IMWEBSRV,ICSPARM='-r /dgv_pet/httpdz0pki.conf'
```

**Result:** The server instance successfully started.

3. We issued the following MVS command to start the second server instance:

```
S IMWEBSRV,ICSPARM='-r /dgv_pet/httpd1443.conf'
```

**Result:** The server instance started but then immediately terminated. The error log showed the following messages:

```
IMW6304E Key Data Base not read. Please check keyfile existence, permission and ownership. Using Default root keys.
IMW6310E SSL support initialization failed, server will run only in non-secure mode without listening on ssl port
IMW0501E Accept soft error: Errno: 121 Errno2: 12b80289 Error: EDC5121I Invalid argument. (errno2=0x12B80289)
IMW0502E Accept () soft error count excessive...shutting down
IMW0375E FastCGI initialization error: failed to bind to AF_INET socket, errno= 1115, errno2= 744c7247.
IMW0375E FastCGI initialization error: failed to bind to AF_INET socket, errno= 1115, errno2= 744c7247.
```

The reason for the IMW6304E, IMW6310E, IMW0501E and IMW0502E messages is because, as we noted earlier in “Running IKYSETUP to perform RACF administration” on page 160, our HTTP servers run under the WEBADM user ID, while the certificate that IKYSETUP generates is owned (by default) by the WEBSRV user ID. To resolve this, we changed the certificate owner to WEBADM.

Also, we run with FastCGI in our environment. FastCGI uses a port defined in the configuration file. Since the two new configuration files were copied from the same file, both had identical FastCGI directives, which means when the second server instance started, it attempted to use the same FastCGI port as the first instance. To resolve this, we commented out the following FastCGI directives in the httpd1443.conf file:

```
ServerInit /usr/lpp/internet/bin/libfcgi.so:FCGIInit /dgv_pet/lgw_fcgi.conf
Service /fcgi-bin/* /usr/lpp/internet/bin/libfcgi.so:FCGIDispatcher*
ServerTerm /usr/lpp/internet/bin/libfcgi.so:FCGIStop
```

We then were able to successfully start the second Web server instance.

## PKI Services

This completes the tasks to update the IBM HTTP Server configuration and start the server instances.

### Tailoring the PKI Services configuration file for LDAP

We followed the instructions in the PKI Services documentation to tailor the PKI Services configuration file, `pkiserv.conf`, for LDAP, but we did encounter one problem:

The instructions state that the value of the `AuthName1` keyword must be set to the same value as the `adminDN` keyword in the `slapd.conf` file. Our `slapd.conf` file contains the following:

```
adminDN "cn=LDAP Administrator, o=Your Company"
```

So, in the `pkiserv.conf` file, we coded the `AuthName1` keyword as follows (using the same case and spacing as in the `slapd.conf` file):

```
AuthName1="cn=LDAP Adminstrator, o=Your Company"
```

When we started the `PKISERVD` daemon (actual step occurs later), we got the following message:

```
IKYL001I Error 82313241 importing LDAP username "CN=LDAP Administrator,o=Your Company":
An OID had an invalid syntax
```

The explanation of the `IKYL001I` message in the PKI Services documentation states, "The OID qualifiers must be specified in uppercase and there cannot be any spaces surrounding the equal signs or commas separating the attribute value assertions (AVAs)." Therefore, we changed the value of the `AuthName1` keyword to make `CN=` and `O=` uppercase and to remove the space after the comma, as follows:

```
AuthName1="CN=LDAP Administrator,O=Your Company"
```

We then restarted the `PKISERVD` daemon and the `IKYL001I` message did not appear.

### Creating VSAM data sets and starting PKI Services

PKI Services requires two VSAM data sets: one for the request database and one for the issued certificate list (ICL). The tasks in this section involve updating and running the `IKYCVSAM` job to create the VSAM data sets, and starting the PKI Services daemon, `PKISERVD`.

#### Updating and running IKYCVSAM to create VSAM data sets

We are attempting to make all of our products SMS-managed in our environment. Therefore, we had to make some changes to the `IKYCVSAM` job that creates the VSAM data sets for PKI Services. We followed the instructions in the PKI Services documentation to copy the `IKYCVSAM` member from `SYS1.SAMPLIB` and update the `JOB` card. To make the VSAM data sets SMS-managed, we also removed all `VOLUME DD` statements, `VOL(xxxxxx)` lines, and `FILE(VOLUME)` lines.

We then submitted the `IKYCVSAM` job. The following messages appeared in the `JESMSGLG` log:

```
IAT6140 JOB ORIGIN FROM GROUP=ANYLOCAL, DSP=IR , DEVICE=INTRDR , 0000
IAT6853 THE CURRENT DATE IS TUESDAY, dd mmm yyyy
IRR010I USERID xxxxxxxx IS ASSIGNED TO THIS JOB.
IAT4401 LOCATE FOR STEP=BLDINDEX DD=BASEDD DSN=PKISRVD.VSAM.OST
IAT4404 DATASET NOT FOUND ON MAIN PROCESSOR Z0
```

```
IAT4401 LOCATE FOR STEP=BLDINDEX DD=AIXDD DSN=PKISRV.D.VSAM.OST.AIX
IAT4404 DATASET NOT FOUND ON MAIN PROCESSOR Z0
IAT4801 JOB IKYCVSAM (JOBnnnnn) EXPRESS CANCELED BY INTERPRETER DSP
```

The job attempts to delete the old data sets before creating them. Since this was the first time we ran the job, there were no old data sets in existence, so the DATASET NOT FOUND messages are not a problem.

We did find that the step to define the alternate index and path was failing with the following error message:

```
GD17103I CATALOG ERROR WHILE DEFINING VSAM DATA SET PKISRV.D.VSAM.OST.AIX
RETURN CODE IS 56 REASON CODE IS 6 IGG0CLFT
```

The problem was that we ran the job from a user ID that did not have the proper authority to perform this step. We reran the job from a user ID that had the necessary authorization and the job ran successfully.

### Starting the PKI Services daemon

We encountered several problems when we attempted to start the PKI Services daemon, PKISERV.D. These were mainly due to configuration problems during the setup tasks above. Although we did not realize the problems until we got to this task and we tried to start the daemon, we've pointed them out in the appropriate preceding sections in order to help readers avoid similar situations.

To preserve a chronological perspective, we'll recap below our experiences with trying to start the PKI Services daemon.

#### ***Experiences with starting the PKI Services daemon:***

- We issued the following command to start the PKI Services daemon:

```
S PKISERV.D
```

The startup failed with the following messages:

```
ICH408I USER(PKISRV.D) GROUP(SYS1) NAME(PKI SRVS DAEMON) CDS.CSSM CL(FACILITY)
INSUFFICIENT ACCESS AUTHORITY
ACCESS INTENT(READ) ACCESS ALLOWED(NONE)
IKYP016I THE PKI SERVICES RUNTIME ENVIRONMENT COULD NOT BE INITIALIZED
ICH408I USER(PKISRV.D) GROUP(SYS1) NAME(PKI SRVS DAEMON) CDS.CSSM CL(FACILITY)
INSUFFICIENT ACCESS AUTHORITY
ACCESS INTENT(READ) ACCESS ALLOWED(NONE)
INSUFFICIENT ACCESS AUTHORITY
ACCESS INTENT(READ) ACCESS ALLOWED(NONE)
```

As we discussed earlier in “Configuring OCSF” on page 158 and “Steps for running IKYSETUP” on page 161, we resolved this by issuing the following RACF commands to give the PKISRV.D user ID READ access to the OCSF data sets and BPX.SERVER facility class:

```
PERMIT CDS.CSSM CLASS(FACILITY) ID(PKISRV.D) ACC(READ)
PERMIT CDS.CSSM.CRYPTO CLASS(FACILITY) ID(PKISRV.D) ACC(READ)
PERMIT CDS.CSSM.DATALIB CLASS(FACILITY) ID(PKISRV.D) ACC(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

- We again issued the S PKISERV.D command. This time we got the following messages:

```
IKYL001I Error 82313241 importing LDAP username "CN=LDAP Administrator,o=Your Company":
An OID had an invalid syntax
```

As we discussed in “Tailoring the PKI Services configuration file for LDAP” on page 166, the explanation of the IKYL001I message indicates that the LDAP

## PKI Services

distinguished name must have all OID qualifiers in uppercase. Accordingly, we changed the specification of the AuthName1 keyword in our pkiserv.conf file to the following:

```
AuthName1="CN=LDAP Administrator,o=Your Company"
```

- We again issued the S PKISERVD command and got the following messages:

```
IKYC009I LDAP post unsuccessful for object id = 100, state =0x2150000, status =6032:
No description found for this error code
IKYP008E Directory post unsuccessful. LDAP data library module rc=6032
IKYC008I Error 5125 creating an CSSM_DL_DB_RECORD_CRL entry for CRL to LDAP for OU=PKI
Certificate Authority,o=Your Company,c=US: No description found for this error code
```

As we noted in “Running IKYSETUP to perform RACF administration” on page 160, the suffix of the PKI Services CA distinguished name must match the LDAP suffix. We had originally defined our LDAP suffix to be “o=SAM Company”, so we also set the o= value for the ca\_dn variable to “SAM Company”. This meant our ca\_dn value was set to “ou=PKI Certificate Authority, o=SAM Company, c=US” but our LDAP suffix was set to “o=SAM Company”. The problem was that by only coding the o= value for the LDAP suffix, it also meant that the ca\_dn hierarchy cannot have anything coded past the o= value. We resolved this by changing the LDAP suffix to “o=SAM Company, c=US” in the slapd.conf file.

- We again issued the S PKISERVD command. We got the following messages:

```
IKYC009I LDAP post unsuccessful for object id = 100, state = 0x2150000, status = 6065:
No description found for this error code
IKYP008E Directory post unsuccessful. LDAP data library module rc = 6065
```

As we discussed in “Steps for tailoring the LDAP configuration for PKI Services” on page 163, the explanation of return code 6065 in *z/OS Open Cryptographic Services Facility Application Programming* is LDAPDL\_OBJECT\_CLASS\_VIOLATION. This means that objects could not be stored in the LDAP directory. After more investigation, it appeared that our schema did not finish loading as we assumed it had.

The problem arose because our original LDAP setup occurred prior to z/OS V1R3 at which time the schema.user.ldif file was loaded into LDAP. The messages that we got earlier while tailoring the LDAP configuration for PKI Services and that we had assumed to be informational really meant that the schema failed to load. The schema failed to load because it was trying to add objects that were already there. APAR OW53820 addresses this problem.

We instead needed to load just the PKI Services information into LDAP by individually loading the following three files:

- EntrustPKIV4.ldif
- EntrustPKIV5.ldif
- RFC2587.ldif

These files reside along with schema.user.ldif in the /usr/lpp/ldap/etc directory. We copied all three files to the /tmp/pki directory and updated the suffix value in the “dn:” line as we had done for the schema.user.ldif file. We issued the following commands from the /tmp/pki directory to load the schema:

```
ldapmodify -h yyy.ibm.com -D "cn=LDAP Administrator,o=Your Company"
-w secret -V 3 -f EntrustPKIV4.ldif
ldapmodify -h yyy.ibm.com -D "cn=LDAP Administrator,o=Your Company"
-w secret -V 3 -f EntrustPKIV5.ldif
ldapmodify -h yyy.ibm.com -D "cn=LDAP Administrator,o=Your Company"
-w secret -V 3 -f RFC2587.ldif
```

**Result:** Each of the above three commands displayed the following response:  
modifying entry cn=schema, o=Your Company.

**Note**

Our experience here differs from the problem conclusion in APAR OW53820. After we concluded our testing, we learned that the closure information for APAR OW53820 now says that only the RFC2587.Idif file and its prerequisites must be loaded.

- We again issued the S PKISERVD command.

**Result:** The PKI Services daemon successfully started.

---

## Customizing PKI Services

Tasks for customizing PKI Services include customizing the end-user Web pages, customizing the administration Web pages, and other advanced customization. The end-user Web pages require some customization in order for certificate processing to work (although this fact wasn't obvious to us from the PKI Services documentation); the other customization tasks are optional. Our testing only included customizing the end-user Web pages.

### Customizing the end-user Web pages

The pkiserv.tmpl certificate templates file and end-user Web pages require *some* level of customization in order for certificate processing to work. We decided to perform the minimal customization as described in the PKI Services documentation. We chose to use the following certificate templates:

- One-year SAF server certificate
- One-year SAF browser certificate
- Two-year PKI browser certificate for authenticating to z/OS
- Five-year PKI SSL server certificate

#### Steps for performing minimal customization

We performed the following steps to update the /etc/pkiserv/pkiserv.tmpl file:

1. For the SAF Server Certificate 1 Year (Auto Approved) certificate, we changed the following named field from this:

```
%%SignWith=SAF:CERTAUTH/taca%%
```

to this:

```
%%SignWith=SAF:CERTAUTH/Local PKI CA%%
```

2. For the SAF Browser Certificate 1 Year (Auto Approved) certificate, we changed the following named fields from this:

```
%%Org=The Firm%%
%%SignWith=SAF:CERTAUTH/taca%%
```

to this:

```
%%Org=SAF through PKI OrgUnit%%
%%SignWith=SAF:CERTAUTH/Local PKI CA%%
```

## PKI Services

3. For the 2 Year PKI Browser Certificate For Authenticating To z/OS certificate, we changed the following named fields from this:

```
%%OrgUnit=Class 1 Internet Certificate CA%%
%%Org=The Firm%%
```

to this:

```
%%OrgUnit=2 Year PKI Browser Cert for Auth to z/OS%%
%%Org=Sue's Company%%
```

- 
4. For the 5 Year PKI SSL Server Certificate, all of the values are optional, so we did not make any changes.
- 

This completes the changes we made to the /etc/pkiserv/pkiserv.tmpl file.

---

## Using PKI Services

We tested the use of the end-user Web pages to work with certificates.

### Using the end-user Web pages

Using the end-user Web pages, we were able to install the CA certificate into the browser and sign certificate requests using both RACF and PKI Services. Below are the steps we followed in each case.

#### Steps for installing the CA certificate into the browser

This was our first time accessing the forms on the user Web pages, so we performed the following steps to install the PKI Services CA certificate into our browser.

1. We went to the URL for our main PKI Services Web page:  

```
http://xxx.ibm.com/PKIServ/public-cgi/camain.rexx
```

---
  2. We clicked the **Install our CA certificate into your browser** link.
    - a. From the File download dialog, we clicked the option to open the file from its current location (rather than save the file to disk).
    - b. On the Certificate dialog window, we clicked the **Install Certificate** button.
    - c. We clicked **Next** on each of the windows that followed, until we got to the one that requested us to specify a short name.
    - d. We entered PKI CA for the short name.
    - e. On the last window, we clicked **Finish**.

---
  3. On the browser, we looked at the security settings and verified that the PKI Services CA certificate was there.
- 

This concludes the one-time setup to install the CA certificate into the browser.

#### Steps for creating a one-year SAF server certificate

We did the following to create a one-year SAF server certificate:

1. We created the Base64 encoded PKCS#10 certificate request through RACF, as follows:
  - a. Issued the following command to create a self-signed server certificate:
 

```
racdcert id(WEBADM) GENCERT SUBJECTSDN(CN('xxx.ibm.com')
 o('SAF Server Cert Org') ou('SAF Server Cert Org Unit')
 L('Poughkeepsie') SP('New York') c('US'))
 SIZE(512) WITHLABEL('Server Cert using SAF')
```
  - b. Issued the following command to generate a server certificate request and copy it to an MVS data set:
 

```
racdcert id(webadm) GENREQ(LABEL('Server Cert using SAF')) DSN(SCERTREQ.ARM)
```
  - c. Downloaded the SCERTREQ.ARM data set into an HFS file.

2. We requested a new certificate through PKI Services, as follows:
  - a. Under the **Request a new certificate using a model** section on the main “PKISERV Certificate Generation Application” Web page, we selected the **1 Year SAF Server Certificate** model from the drop-down list.
  - b. Clicked the **Request Certificate** button.
  - c. The “No User Certificate” page appeared and requested client authentication. We did not have a personal certificate, so we just clicked **OK**.
  - d. On the “SAF Server Certificate 1 Year (Auto Approved)” page, we entered the following values into the fields:

| Field        | Value                    |
|--------------|--------------------------|
| Common Name: | xxx.ibm.com              |
| Org Unit:    | SAF Server Cert Org Unit |
| Org:         | SAF Server Cert Org      |
| Locality:    | Poughkeepsie             |
| State:       | New York                 |
| Country:     | US                       |
| Label:       | Server Cert using SAF    |

**Note:** You do not have to repeat any fields on this Web page that you already supplied in the RACDCERT command.

- e. We copied and pasted the certificate request that was stored in the HFS into the **Base64 encoded PKCS#10 certificate request** field.
- f. Clicked the **Submit certificate request** button.
- g. At the prompt for Surrogate User, we entered the user ID and password and then clicked **OK**.
- h. A new page appeared saying that the request was submitted successfully. The page also contained a transaction ID and a link to the signed server certificate. We clicked the **Continue** button to retrieve the signed certificate.
- i. The “No User Certificate” page appeared and requested client authentication. We did not have a personal certificate, so we just clicked **OK**.
- j. On the “Retrieve Your 1 Year SAF Server Certificate” page, we clicked the **Continue** button.

## PKI Services

- k. The “No User Certificate” page appeared and requested client authentication. We did not have a personal certificate, so we just clicked **OK**.
- l. At the prompt for Authenticated User, we entered the user ID and password and then clicked **OK**.

- 
3. We copied and pasted the signed certificate into an MVS data set, WAJDA.CERT2.ARM.

- 
4. We issued the following commands to add the certificate to RACF and make it the default:

```
RACDCERT ID(WEBADM) ADD('WAJDA.CERT2.ARM') WITHLABEL('Server Cert using SAF')
RACDCERT ID(WEBADM) CONNECT(ID(WEBADM) LABEL('Server Cert using SAF') RING(SSLring) DEFAULT)
```

- 
5. We set the keyfile directive to SSLring in the http.conf file and then restarted the HTTP server.

---

Upon completing the above steps, we were able to successfully make an SSL connection.

### Steps for creating a one-year SAF browser certificate

We did the following to create a one-year SAF browser certificate:

1. Under the **Request a new certificate using a model** section on the main PKI Services Web page, we selected the **1 Year SAF Browser Certificate** model from the drop-down list, then clicked the **Request Certificate** button.

---

2. The “No User Certificate” page appeared and requested client authentication. We did not have a personal certificate, so we just clicked **OK**.

---

3. On the “SAF Browser Certificate 1 Year (Auto Approved)” page, we did the following:
  - a. In the **Label** field, entered: SAF Browser Cert/ PKI Services
  - b. Selected a key size of 1024 (High Grade)
  - c. Clicked the **Submit certificate request** button.

---

4. On the “Generate A Private Key” page, we clicked **OK**.

---

5. On the “No User Certificate” page, we clicked **OK**.

---

6. At the prompt for SurrogateUser, we entered the user ID and password and then clicked **OK**.

---

7. Again, on the “No User Certificate” page, we clicked **OK**.

---

8. A new page appeared saying that the request was submitted successfully. The page also contained a transaction ID and a link to the signed server certificate. We clicked the **Continue** button to retrieve the signed certificate.

---
9. On the “Retrieve Your 1 Year SAF Browser Certificate” page, we clicked the **Retrieve and Install Certificate** button.  
**Result:** After processing for a moment, the message Done appeared.

---

To verify that the certificate was successfully installed, we viewed the security settings for our browser and saw that the new certificate was there. We were then able to use the browser certificate to successfully make a connection.

### Steps for creating a two-year PKI browser certificate for authenticating to z/OS

We did the following to create a two-year PKI browser certificate for authenticating to z/OS:

1. Under the **Request a new certificate using a model** section on the main PKI Services Web page, we selected the **2 Year PKI Browser Certificate For Authenticating To z/OS** model from the drop-down list, then clicked the **Request Certificate** button.

---
2. On the “2 Year Browser Certificate For Authenticating to z/OS” page, we did the following:
  - a. Entered a pass phrase for securing this request (and then re-entered it for confirmation)
  - b. Clicked **OK** on the “Generate A Private Key” page
  - c. Selected a key size of 512 (Low Grade)
  - d. Clicked the **Submit certificate request** button.

---
3. On the “Generate A Private Key” page, we clicked **OK**.

---
4. At the prompt for SurrogateUser, we entered the user ID and password and then clicked **OK**.

---
5. A new page appeared saying that the request was submitted successfully. The page also contained a transaction ID and a link to the new browser certificate. We clicked the **Continue** button to retrieve the new certificate.

---
6. On the “Retrieve Your 2 Year PKI Browser Certificate For Authenticating to z/OS” page, we entered the pass phrase and then clicked the **Retrieve and Install Certificate** button.  
**Result:** After processing for a moment, the message Done appeared.

---

To verify that the certificate was successfully installed, we viewed the security settings for our browser and saw that the new certificate was there. We were then able to use the browser certificate to successfully make a connection.

## Steps for creating a five-year PKI SSL server certificate

We did the following to create a five-year PKI SSL server certificate:

1. We created the Base64 encoded PKCS#10 certificate request through RACF, as follows:
  - a. Issued the following command to create a self-signed server certificate:
 

```
RACDCERT ID(WEBADM) GENCERT SUBJECTSDN(CN('xxx.ibm.com')
o('5 Year PKI SSL Cert org') ou('PKI SSL Cert Org')
L('Poughkeepsie') SP('New York') c('US'))
SIZE(512) WITHLABEL('Server Cert using PKI')
```
  - b. Issued the following command to generate a server certificate request and copy it to an MVS data set:
 

```
RACDCERT ID(WEBADM) GENREQ(LABEL('Server Cert using PKI')) DSN(PCERTREQ.ARM)
```
  - c. Downloaded the PCERTREQ.ARM data set into an HFS file.

2. We requested a new certificate through PKI Services, as follows:
  - a. Under the **Request a new certificate using a model** section on the main PKI Services Web page, we selected the **5 Year PKI SSL Server Certificate** model from the drop-down list.
  - b. Clicked the **Request Certificate** button.
  - c. On the “5 Year PKI SSL Server Certificate” page, we entered the following values into the fields:

| Field        | Value                   |
|--------------|-------------------------|
| Common Name: | xxx.ibm.com             |
| Org Unit:    | PKI SSL Cert Org        |
| Org:         | 5 Year PKI SSL Cert org |
| Locality:    | Poughkeepsie            |
| State:       | New York                |
| Country:     | US                      |

- d. We copied and pasted the certificate request that was stored in the HFS into the **Base64 encoded PKCS#10 certificate request** field.
  - e. We entered a pass phrase to secure the request.
  - f. Clicked the **Submit certificate request** button.
  - g. A new page appeared saying that the request was submitted successfully. The page also contained a transaction ID and a link to the signed server certificate. We clicked the **Continue** button to retrieve the signed certificate.
  - h. On the “Retrieve Your 5 Year PKI SSL Server Certificate” page, we entered the pass phrase and clicked **Continue**.

3. We copied and pasted the signed certificate into an MVS data set, WAJDA.PKICERT.ARM.

4. We issued the following commands to add the certificate to RACF and make it the default:

```
RACDCERT ID(WEBADM) ADD('WAJDA.PKICERT.ARM') WITHLABEL('PKI Cert Req 1')
RACDCERT ID(WEBADM) CONNECT(ID(WEBADM) LABEL('PKI Cert Req 1') RING(SSLring) DEFAULT)
```

- 
5. We set the keyfile directive to SSLring in the http.conf file and then restarted the HTTP server.
- 

Upon completing the above steps, we were able to successfully make an SSL connection.

## PKI Services

---

## Chapter 14. Using IBM WebSphere Application Server for z/OS and OS/390

This chapter describes our experiences using IBM WebSphere Application Server Standard Edition for OS/390 V3.5 (hereafter referred to as "Standard Edition V3.5" or simply "SE V3.5") and versions 4.0 and 4.0.1 of IBM WebSphere Application Server for z/OS and OS/390 (hereafter referred to as "WebSphere for z/OS V4.x" or simply "V4.x").

There is also a Web-based user interface, the *WebSphere Troubleshooter for OS/390*, which contains the most current debugging and tuning hints for all supported releases of IBM HTTP Server and IBM WebSphere Application Server. You can find the Troubleshooter at:

<http://www.ibm.com/software/webservers/appserv/troubleshooter.html>

In this chapter, we discuss the following topics:

- "Packaging and deploying Web applications using WAR files"
- "Migrating Web applications to WebSphere Application Server V4.0 for z/OS and OS/390" on page 180
- "Enabling WebSphere Application Server V4.0 for z/OS and OS/390 on a sysplex" on page 181
- "Updating the configuration for servlets that access MOFW or EJB applications from WebSphere for z/OS V4.0 plugin" on page 187
- "Migrating to WebSphere Application Server V4.0.1 for z/OS and OS/390" on page 188

---

### Packaging and deploying Web applications using WAR files

In this section, we share our experience with one of the new features of Standard Edition V3.5: the ability to deploy Web applications that have been packaged into a Web ARchive format (WAR) file.

WAR files were introduced as part of the Java Servlet 2.2 API Specification and provide a way to structure Web applications for deployment. They also help strengthen the separation between the development and deployment tasks. See the Java Servlet 2.2 API Specification at <http://java.sun.com/products/servlet/> for complete details.

### Steps for creating and deploying WAR files

The following steps describe the high-level process that we used to create and deploy our Web application using WAR files. We offer additional comments about this process in the next section, "Experiences with WAR files" on page 178.

1. Create the Web application in WebSphere Studio. (We used VisualAge for Java to create the servlets and imported them into WebSphere Studio.)
2. Set the appropriate **Webapp web path** for the server in the publishing properties in the publishing view. This will serve as the Root URI for the Web application.
3. Create the **Web Configuration Descriptor file** in WebSphere Studio.

## WebSphere Application Server

4. Update the Web configuration descriptor file.
5. Create the WAR file in WebSphere Studio.
6. Publish the WAR file to our z/OS system.
7. Run the *WebSphereApplicationServer\_InstallDirectory/bin/wartowebapp.sh* script from the directory that contains the WAR file on the z/OS system.
8. Add the **webapp** properties created by the *wartowebapp.sh* script to the *was.conf* file for the WebSphere Application Server on which the application is to be deployed.
9. Stop and restart the HTTP Server configured with the WebSphere Application Server to pick up the changes.

The WAR file is now deployed on WebSphere Application Server on the z/OS system.

## Experiences with WAR files

Because WAR files are essentially JAR files with a defined structure, any standard Java archive tool can create them. We used WebSphere Studio V3.5.3 on our Windows workstation to help create and publish our WAR files. WebSphere Studio makes this process easy.

We had previously been using WebSphere Studio to create and publish our Web applications and we found that converting them to WAR files only simplifies things. Since information about the Web application is conveyed between the application developer and the deployer by way of the deployment descriptor file contained within the WAR file, structuring our application in WebSphere Studio was much easier. Previously, we would set up our directories in WebSphere Studio to mimic the directories where the application would ultimately be installed. Using WAR files, however, WebSphere Studio places the files in the appropriate directories within the WAR file and updates the links within the files. Therefore, the placement of files in directories inside WebSphere Studio is not as important; it's now more of a matter of preference to the developer.

We did find that packaging Web applications in WAR files is best used for complete and fully tested applications. While we were developing our applications, we found it faster to manually configure the test application server and publish as needed. With the Web application configured for autoreload, we could update a portion of the Web application, publish it, and see the changes almost immediately. Once we were satisfied that our application was fully ready, we then packaged it into a WAR file. We then used the WAR file to rapidly deploy the complete application on multiple systems.

While WebSphere Studio helps to initially create the deployment descriptor for the WAR file, we found that we had to manually add many of the available elements,

such as the context parameters, welcome file list, and error page. (Refer to the Java Servlet 2.2 Specification for a full list of available elements.)

Figure 18 contains a sample of one of the deployment descriptors that we used with the additional elements added:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.2//EN"
"http://java.sun.com/j2ee/dtds/web-app_2_2.dtd">
<web-app>
<display-name>TestWebApp1</display-name>
<description>Test Web Application #1</description>
<distributable></distributable>
<context-param>
 <param-name>ContextParm1</param-name>
 <param-value>This is the value of ContextParm1</param-value>
</context-param>
<servlet>
 <servlet-name>simpleMsgServletModel</servlet-name>
 <display-name>simpleMsgServletModel Display Name</display-name>
 <description>Simple Message Servlet Model Servlet</description>
 <servlet-class>ebIT.Test.TestWebApp1.simpleMsgServletModel</servlet-class>
</servlet>
<servlet>
 <servlet-name>ebIT.Test.TestWebApp1.simpleMsgServletModel</servlet-name>
 <display-name>Simple Message Servlet Model Display Name</display-name>
 <description>Simple Message Servlet Model Servlet</description>
 <servlet-class>ebIT.Test.TestWebApp1.simpleMsgServletModel</servlet-class>
</servlet>
<servlet>
 <servlet-name>simpleMsgSrvltMdl</servlet-name>
 <display-name>simpleMsgSrvltMdl Display Name</display-name>
 <description>Simple Message Servlet Model Servlet</description>
 <servlet-class>ebIT.Test.TestWebApp1.simpleMsgServletModel</servlet-class>
 <init-param>
 <param-name>simpleMsgSvltMdlInitParm1</param-name>
 <param-value>some_servlet_init_parm</param-value>
 </init-param>
</servlet>
<servlet-mapping>
 <servlet-name>ebIT.Test.TestWebApp1.simpleMsgServletModel</servlet-name>
 <url-pattern>/servlet/ebIT.Test.TestWebApp1.simpleMsgServletModel</url-pattern>
</servlet-mapping>
<servlet-mapping>
 <servlet-name>simpleMsgServletModel</servlet-name>
 <url-pattern>/servlet/simpleMsgServletModel</url-pattern>
</servlet-mapping>
<session-config>
 <session-timeout>10</session-timeout>
</session-config>
<welcome-file-list>
 <welcome-file>index.html</welcome-file>
 <welcome-file>Welcome.html</welcome-file>
</welcome-file-list>
<error-page>
 <error-code>404</error-code>
 <location>/error404.jsp</location>
</error-page>
</web-app>
```

Figure 18. Sample of one of our deployment descriptor files

## WebSphere Application Server

Standard Edition V3.5 does not actually consume the WAR files; it just uses them as a deployment vehicle. The `wartowebapp.sh` script that WebSphere provides unpacks the WAR file and creates a file containing the appropriate **webapp** properties. These properties must then be added to the application server's configuration file (`was.conf`) to actually deploy the Web application (see step 8 earlier in "Steps for creating and deploying WAR files" on page 177).

**Tip:** If your Web application was previously deployed on WebSphere Application Server and you intend to redeploy it using a WAR file and want to use the same **webapp** name (as we initially did during our testing), make sure you comment out any **webapp.webapp-name.** type properties in your `was.conf` file. If WebSphere Application Server finds any **webapp.** properties, it will not search the deployed Web application's classpath for the XML file containing the configuration definitions created by the `wartowebapp.sh` script.

---

## Migrating Web applications to WebSphere Application Server V4.0 for z/OS and OS/390

In this section, we share our experiences with migrating our Web applications from Standard Edition V3.5 to WebSphere for z/OS V4.0.

We describe our initial migration of Web applications that only consist of Java servlets, JavaServer Pages (JSPs), and static Web content to the V4.0 plugin. The V4.0 plug-in environment does not directly support Enterprise JavaBeans (EJB) beans; however, it does support Web applications that access EJB beans running in a WebSphere for z/OS V4.0 J2EE server.

The book *WebSphere Application Server V4.0.1 for z/OS and OS/390: Assembling J2EE Applications*, SA22-7836, provides extensive coverage of the various migration paths. We recommend that you carefully review this book when planning your migration. (See "Where to find more information" on page 192.)

## Migrating Web applications to the WebSphere for z/OS V4.0 plugin

We found our initial migration to WebSphere for z/OS V4.0 to be easy. Although the overall V4.0 product changes dramatically from previous versions, we would like to highlight the following points that helped to simplify our initial migration:

- V4.0 and SE V3.5 can coexist on the same system as long as they are installed into separate directories.
- The Go Webserver (GWAPI) plug-in routine in WebSphere for z/OS V4.0 continues to run in the same address space as the HTTP server.
- The plug-in routine in V4.0 works identically to the one in SE V3.5.
- Applications that run on SE V3.5 can run unchanged in the V4.0 plugin.

For the above reasons, we did not have to initially deploy our Web applications into a J2EE server (where they would have to fully comply with the Servlet 2.2 API and JSP 1.1 specifications and be packaged in EAR files, as we describe in the next section) in order to begin using the WebSphere for z/OS V4.0 environment. By installing each WebSphere version into separate directories, we were able to continue running our production workloads using Standard Edition V3.5 on one set of servers while we began experimenting with the same applications using WebSphere for z/OS V4.0 on another set of servers.

**Configuring the V4.0 plug-in environment:** The environment of the V4.0 plugin is very familiar—we only needed to change the **appserver.version** property in the

## WebSphere Application Server

application server's was.conf file. Similarly, to configure the HTTP server, we only had to update the **ServerInit**, **ServerTerm**, and **Service** directives in the httpd.conf file to point to the appropriate Websphere level, and update the NLSPATH in the httpd.envvars file.

Figure 19 shows two of our HTTP server instances (address spaces): one with Standard Edition V3.5 and the other with the WebSphere for z/OS V4.0 plugin:

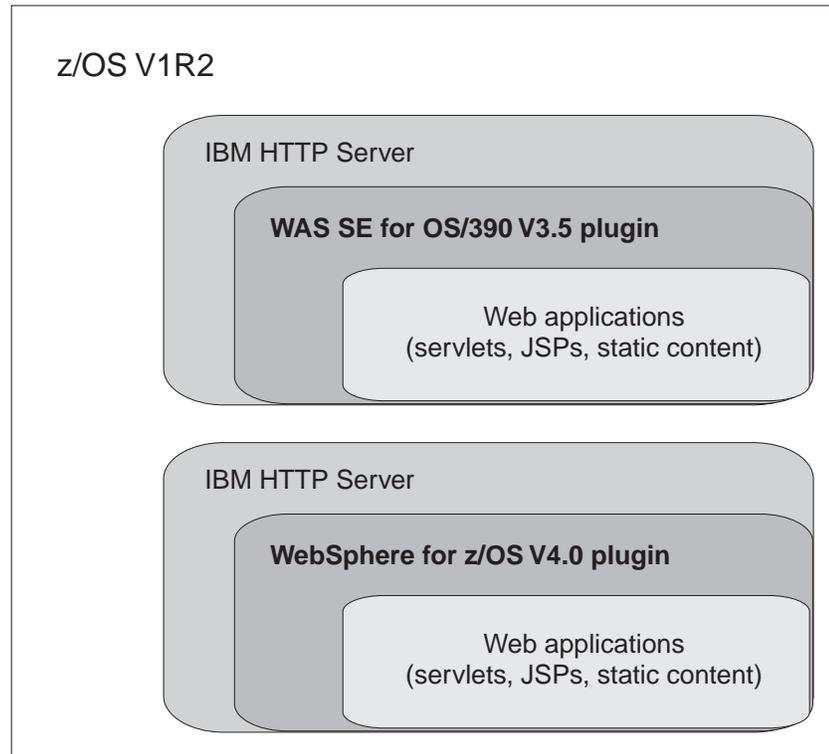


Figure 19. Separate HTTP server instances with WAS SE V3.5 and WebSphere for z/OS V4.0 plugins

---

## Enabling WebSphere Application Server V4.0 for z/OS and OS/390 on a sysplex

In this section, we continue to describe our experiences with WebSphere Application Server V4.0 for z/OS and OS/390 that began in the April 2001 edition of *OS/390 e-business Integration Test (ebIT) Report*, SA22-7461-04. That edition is available on the Library page of the z/OS Integration Test Web site at <http://www.ibm.com/servers/eserver/zseries/zos/integtst/>.

We now explore setting up a WebSphere host cluster by configuring WebSphere for z/OS V4.0 on three z/OS images (JF0, JB0, J80) in the sysplex. The benefits of migrating to a host cluster in a sysplex environment are:

- You can balance the workload across multiple systems, thus providing better performance management for your applications.
- As your workload grows, you can add new systems to meet demand, thus providing a scalable solution to your processing needs.

## WebSphere Application Server

- By replicating the run time and associated business application servers, you provide the necessary system redundancy to assure availability for your users. In the event of a failure on one system, you have other systems available for work.

The following sections closely parallel those found under the "Enabling WebSphere for z/OS on a sysplex" section in the "Advanced topics" chapter of *WebSphere Application Server V4.0.1 for z/OS and OS/390: Installation and Customization*, GA22-7834. Note that we do not reiterate verbatim the instructions from that book and we perform some steps in a slightly different order. Therefore, the following information should accompany—not replace—your reading of the corresponding sections of *Installation and Customization*.

The following information describes what we did to add the second system (JB0) to our WebSphere host cluster. We then repeated the process to add the third system (J80).

## Planning for WebSphere for z/OS and sysplex

Since we already installed and configured a set of WebSphere for z/OS run-time and business application servers on a single z/OS system in our sysplex, we made the following planning decisions to prepare to add another system image to our WebSphere host cluster:

- We decided to have a single-system view of the error log when we first installed WebSphere for z/OS on a single system and we configured it to use the coupling facility.
- We used a shared HFS in read/write mode across the sysplex for WebSphere for z/OS environment files.
- We used a shared PROCLIB for all servers.
- We decided to replicate the Naming Server, System Management Server, and Interface Repository Server on each host cluster instance.

## Preparing the security system

We did the following to set up the z/OS Security Server (RACF):

1. We used a shared RACF database, thus Websphere or z/OS assumes that a given user ID represents the same user identity on all systems in the sysplex.

---

2. We used shared start procedures and defined the user identity to RACF once through the STARTED class. We granted other RACF authorizations to the user identities that we defined in the STARTED class. Thus, replicated control regions and server regions run under the same user ID and have the same authorizations throughout the sysplex.

---

## Setting up data sharing

We set up DB2 data sharing according to the instructions in *Installation and Customization*.

## Customizing base z/OS functions on other systems in the sysplex

We did the following to customize the base z/OS functions on other systems in the sysplex:

1. Since we use a shared parmlib and proclib for our sysplex, we did not have to make changes to the base system.

2. We dynamically added data sets *hlq.SBBOLOAD* and *hlq.SBBOLPA* into the LPA by issuing the following commands on each system in our WebSphere host cluster:

```
SETPROG LPA,ADD,MASK=*,DSNAME=hlq.SBBOLOAD
SETPROG LPA,ADD,MASK=*,DSNAME=hlq.SBBOLPA
```

3. We made sure that data set *hlq.SBBOLD2* was in the linklist. We actually added it to the linklist and APF authorization during the installation and configuration of our first image. Because we use shared system data sets, these changes are picked up by all systems in the sysplex.
4. In the above examples, we used a high-level qualifier (*hlq*) of *EBIT.WAS40.GA*. Thus, we made sure that all WebSphere data sets (*EBIT.WAS40.GA.\**) and LDAP data sets (*GLD.\**) were cataloged.

## Updating the TCP/IP configuration

We did the following to update our TCP/IP configuration:

1. We updated our DNS to reflect our systems' names and IP addresses. We also created an entry for the Generic Daemon IP name.
2. In the TCP/IP profile, we added port number 900 and associated it with a new System Management server instance name, and we added port number 6666 and associated it with a new Daemon server instance name.

### Example:

```
900 TCP SYSMGT02
6666 TCP DAEMON02
```

3. We updated the hosts file on our Windows NT workstation, as follows:

```
xx.xx.xx.37 petwas petwas.xxx.ibm.com
xx.xx.xx.37 jf0eip.jf0eip.xxx.ibm.com
xx.xx.xx.37 jf0eip jf0eip.xxx.ibm.com
#
xx.xx.xx.33 jb0eip jb0eip.xxx.ibm.com
xx.xx.xx.33 jb0eip.jb0eip.xxx.ibm.com
#
xx.xx.xx.30 j80eip j80eip.xxx.ibm.com
xx.xx.xx.30 j80eip.j80eip.xxx.ibm.com
```

## Setting up LDAP files for other systems in the sysplex

We did the following to set up the LDAP files for another system:

1. We copied the LDAP files from our initial setup of system JF0 in our */WebSphere390/CB390/UTCPLXJ8/etc/ldap* directory to system JB0, as follows:
  - copied *JF0.dsnaoini* to *JB0.dsnaoini*
  - copied *JF0.bboslafd.conf* to *JB0.bboslafd.conf*

## WebSphere Application Server

- copied JF0.bboidif.cb to JB0.bboidif.cb

- 
2. We also updated the JB0.dsnaoini file with our DB2 subsystem name, DBB1.
- 

## Defining the second WebSphere for z/OS system

We used the IBM WebSphere for z/OS Administration application from our Windows NT workstation to define the second z/OS system in our three-way WebSphere host cluster. We then repeated the same process to define the third system.

### Steps to define our second WebSphere for z/OS system

**Before we began:** We started our initial WebSphere for z/OS daemon and associated servers on our first z/OS system, as follows:

```
S BBODMN.DAEMON01
S BBOASR1.BBOASR1A
S BBOASR2.BBOASR2A
```

We then did the following to add the second system:

1. On the NT workstation, we logged on to the IBM WebSphere for z/OS Administration application.

- 
2. We added a new conversation by clicking **Conversation→Add** and entered the name of the conversation.
- 

3. We added the second system in the sysplex by double-clicking the new conversation name then clicking **Sysplexes→Sysplex Name (UTCPLXJ8)**. We right-clicked **Systems**, clicked **Add**, entered JB0 as the name of the system, then clicked the **Save** (diskette) icon.

This action creates all of the run-time server instances, namely DAEMON02, SYSMGT02, INTFRP02, and NAMING02.

- 
4. Next, we updated the environment variables for the DAEMON02 server instance, as follows:

- Double-clicked **Systems→JB0→Server Instances→DAEMON02**.
- In the list of environment variables, we added the following values:

```
DM_SPECIFIC_SERVER_NAME=DAEMON02
IR_SPECIFIC_SERVER_NAME=INTFRP02
NM_SPECIFIC_SERVER_NAME=NAMING02
SM_SPECIFIC_SERVER_NAME=SYSMGT02
SYS_DB2_SUB_SYSTEM_NAME=DBB1
SMPROC=BBOSMS
NMPROC=BBONM
IRPROC=BB0IR
```

- We then clicked the **Save** (diskette) icon.

- 
5. Using the same procedure, we updated the environment variables for the SYSMGT02, NAMING02, and INTFRP02 server instances with the following values:

```
DM_SPECIFIC_SERVER_NAME=DAEMON02
IR_SPECIFIC_SERVER_NAME=INTFRP02
NM_SPECIFIC_SERVER_NAME=NAMING02
SM_SPECIFIC_SERVER_NAME=SYSMGT02
SYS_DB2_SUB_SYSTEM_NAME=DBB1
```

---

6. We then added application server instances, as follows:
- Double-clicked our new conversation name then clicked **Sysplexes>Sysplex Name (UTCPLXJ8)>Systems>JB0>Server instances**
  - Right-clicked **Server Instance** then clicked **Add**
  - Entered BBOASR1B for the **Server Instance name**
  - Selected server BBOASR1
  - Set the following environment variables:

```
DM_SPECIFIC_SERVER_NAME=DAEMON02
IR_SPECIFIC_SERVER_NAME=INTFRP02
NM_SPECIFIC_SERVER_NAME=NAMING02
SM_SPECIFIC_SERVER_NAME=SYSMGT02
SYS_DB2_SUB_SYSTEM_NAME=DBB1
```

We repeated this procedure to add the BBOASR2B server instance.

---

7. Next, we updated the following Logical Resource Mapping (LRM) instances:

```
CB_OS/390_Base_DB2_JB0
CB_OS/390_Lifecycle_DB2_JB0
CB_OS/390_Naming_DB2_JB0
CB_OS/390_SysMgt_DB2_JB0
CB_OS/390_Repository_DB2_JB0
```

We updated the DB2 subsystem name for each LRM, as follows:

- Double-clicked our new conversation name then clicked **Sysplexes>Sysplex Name (UTCPLXJ8)>Systems>JB0>LRM Instance**
  - Selected one LRM instance at a time and updated its DB2 subsystem name to DBB1
- 

8. We created a new LRM instance for the installation verification program (IVP), as follows:

- Double-clicked our new conversation name then clicked **Sysplexes>Sysplex Name (UTCPLXJ8)>Systems>JB0**
  - Right-clicked **LRM Instance** then clicked **Add**
  - Entered CB\_OS/390\_IVP\_DB2\_JB0 for the LRM instance name
  - Set the CollectionId to CBIVP\_PKG and set the DB2 subsystem name to DBB1
  - We clicked the **Save** (diskette) icon.
- 

9. We also created a new J2EE resource instance for system JB0, as follows:

- Clicked **J2EE Resources>BBOASR2\_EJB\_IVP\_RESOURCE**
- Right-clicked **J2EE Resource Instance** and clicked **Add Instance**
- We entered the following values:

```
J2EE Resource
Instance name: BBOASR2_EJB_IVP_RESOURCE_JB0
```

## WebSphere Application Server

Database name: USIBMT6PETDB2  
DB2 Plan name: DSNJDBC  
Log stream name: PET.WAS

- Clicked the **Save** (diskette) icon

---

10. To validate our new conversation, we right-clicked the conversation and then clicked **Validate**.

---

11. To commit our new conversation, we right-clicked the conversation and then clicked **Commit**.

---

12. To complete all tasks, we right-clicked our conversation and then clicked **Complete→All tasks**.

---

13. We again clicked **Complete→All tasks** to mark all tasks complete.

---

14. We then activated the new conversation by right-clicking the conversation then clicking **Activate**.

---

15. We started the daemon and other servers (including the application servers, BBOASR1B and BBOASR2B) on system JB0, as follows:

```
S BBODMN.DAEMON02,SRVNAME='DAEMON02'
S BBOASR1.BBOASR1B,SRVNAME='BBOASR1B'
S BBOASR1.BBOASR2B,SRVNAME='BBOASR2B'
```

---

Once all the servers successfully started, our second WebSphere for z/OS system was operational. We proceeded to run the installation verification program.

## Running the installation verification programs (IVP)

We did the following to run the installation verification programs:

1. We submitted the IVP job, BBOIVP, from our newly configured WebSphere system JB0. All three steps of job BBOIVP successfully completed.

---

2. We ran the EJB IVP job, BBOIVPE. It also successfully completed.

---

3. We then stopped the BBOASR1B and BBOASR2B server instances on system JB0 but made sure they were still running on our first WebSphere system, JF0.

---

4. We then submitted the BBOIVP job again from system JB0. Again, all three steps of the IVP successfully ran, but we did receive the following warning message:

```
BossLog: { 0001} 2001/06/29 19:33:16.536 01 SYSTEM=JB0 CLIENT=USER1 PID=0X03
B0050 TID=0X27750800 00000000 c=UNK ./bbossior.cpp+3205 ...
BBOU0706W Invalid EM_USERID or invalid REM_PASSWORD defined as an
environmental variable
Trace: 2001/06/29 19:33:16.553 01 t=7E33C8 c=UNK key=P8 (0000000A)
Description: Log Boss/390 Error
from filename: ./bbossior.cpp
at line: 3205
error message: BBOU0706W Invalid REM_USERID or invalid REM_PASSWORD defined
as an environmental variable
```

We were set up to run with a nonauthenticated client, so we reported this problem to IBM support. APAR PQ50486 addresses this problem. In the meantime, you can circumvent this problem by specifying a valid REM\_USERID and REM\_PASSWORD in a current.env file that the BBOIVP client has permission to read.

- 
5. We stopped the daemon and servers on system JB0 and submitted the BBOIVP job again. It ran fine, but we did receive the following error message:

```
BossLog: { 0001} 2001/07/18 14:12:03.039 01 SYSTEM=JB0 CLIENT=USER1 PID=0X000
B0029 TID=0X27747800 00000000 c=UNK ./bbossecm.cpp+3642 ...
BBOU0507E MSG_BBOUE NUS_SEC_PTKT_CREATE_FAILED: Security manager: pass ticket
create failed
Trace: 2001/07/18 14:12:03.053 01 t=7E3E88 c=UNK key=P8 (0000000A)
Description: Log Boss/390 Error
from filename: ./bbossecm.cpp
at line: 3642
error message: BBOU0507E MSG_BBOUENUS_SEC_PTKT_CREATE_FAILED:
Security manager: pass ticket create failed
```

We had configured the BBOASR1 server to accept nonauthenticated clients as well as basic userID and password authentication. We reported the problem to IBM support. APAR PQ50858 addresses this problem.

---

This completes the installation verification programs.

---

## Updating the configuration for servlets that access MOFW or EJB applications from WebSphere for z/OS V4.0 plugin

In the April 2001 edition of *OS/390 e-business Integration Test (ebIT) Report*, SA22-7461-04, we described our experience with creating and running a servlet that accesses an EJB application. We also gave examples of how to update the classpath and libpath in the was.conf configuration file.

We revisit that topic now to highlight some additional updates that we made to the HTTP server's startup procedure and to the httpd.conf, httpd.envvars, and was.conf configuration files.

## Updating the startup procedure for the HTTP server

We added the following steplib to the startup procedure for the HTTP server:

```
STEPLIB=hlq.SBB0ULIB
```

We use a high-level qualifier (*hlq*) of EBIT.WAS40.GA.

## WebSphere Application Server

### Updating the httpd.conf file

We added the following directives to our httpd.conf file:

```
Pass /images/* /http_shared/images/*
Pass /*.html /http_shared/html/*
```

### Updating the httpd.envvars file

We made the following changes to our httpd.envvars file:

```
LIBPATH=/usr/lpp/internet/bin:/usr/lpp/internet/sbin:/usr/lpp/ldap/lib:/usr/lpp/
WebSphere40/lib:/db2710/current/lib
REM_USERID=CBIVP
REM_PASSWORD=CBIVP
```

### Updating the was.conf file

We made the following changes (shown in bold type) to our was.conf file:

```
appserver.libpath=/usr/lpp/ctg/current/bin:/db2710/current/lib:/usr/lpp/WebSphere
e40/lib:

appserver.classpath=/db2710/current/classes:/db2710/current/classes/db2j2classes.zip:
/PARTAPP/partam02.jar:/HELLOWORLD/HelloWorld.jar:/usr/lpp/WebSphere40/lib/ws390crt.jar:
/usr/lpp/ldap/lib/ibmjndi.jar:/usr/lpp/WebSphere40/samples/PolicyIVP/jcivp.jar:
/usr/lpp/WebSphere40/samples/PolicyIVP/PRODUCTION/bbopl.c.jar

CBIVP and EJB part application servlets
deployedwebapp.cbivp.host=default_host
deployedwebapp.cbivp.rooturi=/webapps/cbivp
deployedwebapp.cbivp.classpath=/JF0/wa/webapps/cbivp/servlets
deployedwebapp.cbivp.documentroot=/JF0/wa/webapps/cbivp/web
deployedwebapp.cbivp.java.naming.factory.initial=com.ibm.ws.naming.WsnInitialContextFactory
deployedwebapp.cbivp.javax.naming.provider.url="iiop://9.xx.xx.xx:900"
deployedwebapp.cbivp.autoreloadinterval=5000
webapp.cbivp.description=CBIVP WebApp
webapp.cbivp.servletmapping=/servlet/*
webapp.cbivp.filemapping=*.html
webapp.cbivp.filemapping=*.gif
```

### Accessing our servlet

After recycling our Web server, we accessed our part02am.html file from the browser, which displayed the form. We filled out the form and clicked the **Send to Server for Execution** button which invokes our servlet that accesses an EJB application running on WebSphere for z/OS V4.0. The servlet ran successfully.

**Hints and tips:** We have the following hints and tips to share:

- Migrate your WebSphere for z/OS V4.0 to at least PTF UQ58062 and UQ56513.
- On z/OS V1R2, apply the fix for APAR OW50714 for LDAP.

---

## Migrating to WebSphere Application Server V4.0.1 for z/OS and OS/390

This section describes our experiences with migrating from WebSphere for z/OS V4.0 to WebSphere for z/OS V4.0.1. Currently, we have completed the migration steps; however, we are still testing the new level and have not yet moved into a "production" mode. We will continue to report on our test experiences in the next edition of our test report.

We used the information in *WebSphere Application Server V4.0.1 for z/OS and OS/390: Migration* (GA22-7860) and *WebSphere Application Server V4.0.1 for z/OS and OS/390: Installation and Customization* (GA22-7834, specifically at the -02 level), available on the Library page of the WebSphere Application Server for z/OS and OS/390 Web site.

The chapter in *Installation and Customization* about installing new releases and maintenance levels describes several methods you can use to install a new functional level of WebSphere for z/OS. They are:

- Cold start
- Warm start
- Hot start
- Quick start

We chose the warm start method for our migration, which we describe below.

We performed our migration using the following software levels on our sysplex:

- z/OS V1R3
- DB2 UDB for OS/390 and z/OS V7.1 and JDBC driver with APAR PQ54756 (PTF UQ61107)
- IBM Developer Kit for OS/390, Java 2 Technology Edition, SDK level 1.3 with APAR PQ57514 (PTF UQ62958)
- WebSphere Application Server V4.0.1 for z/OS and OS/390, PTF UQ62183 and PTF UQ62923
- WebSphere Application Server Standard Edition for OS/390 V3.5, PTF UQ62555

### Steps to migrate from WebSphere for z/OS V4.0 to V4.0.1

We followed the steps for a warm start as described in *WebSphere Application Server V4.0.1 for z/OS and OS/390: Installation and Customization* and followed with the dynamically generated migration instructions.

**Note:** The steps in *Installation and Customization* explain how to run the WebSphere for z/OS Customization ISPF dialog. The outputs of the dialog include a set of tailored migration jobs and a further set of instructions to run those jobs and complete the migration.

**Before we began:** We verified the following information before we started our migration activities:

- We had the appropriate level of service installed on our WebSphere for z/OS V4.0 system, according to the information in *WebSphere Application Server V4.0.1 for z/OS and OS/390: Migration* and the PSP bucket.
- We had our display set up to support a minimum screen size of 32 rows by 80 columns (32x80) in order to run the WebSphere for z/OS Customization ISPF dialog.
- We also made sure that the ISPF split-screen function was not active. (The split-screen function obscures lines at the bottom of the customization dialog panels.)

We then performed the following steps for the warm-start method:

1. We used SMP/E to install the new WebSphere for z/OS V4.0.1 code into new (separate from V4.0) MVS and HFS data sets.  
As part of this step, we also performed the following actions:
  - a. Cataloged the new data sets
  - b. Updated parmlib to APF-authorize the new data sets
  - c. Added the *hlq.SBBOLD2* data set to the linklist
  - d. Dynamically added and deleted modules in the LPA from data sets *hlq.SBBOLOAD* and *hlq.SBBOLPA*
  - e. Mounted the new HFS data set at our previously-used mount point, */usr/lpp/WebSphere40*

2. We backed up our current system, including:
    - The system management database
    - The LDAP database tables that contain the naming space and the interface repository
    - Files in the HFS containing WebSphere for z/OS run-time information (mounted at /WebSphere390/CB390)
    - WebSphere for z/OS PROCLIBs
    - WebSphere for z/OS LOADLIBs
- 

3. Next we ran the WebSphere for z/OS Customization ISPF dialog. To start the dialog, we entered the following from the ISPF command line:

```
ex 'ebit.was401.sbboclib(bbowstrt)' 'hlq(ebit.was401)'
```

We chose a high-level qualifier (HLQ) of EBIT.WAS401 for the SBBOCLIB data set and WebSphere for z/OS product data sets.

In the dialog, we performed the following actions:

- a. We pressed Enter to proceed past the initial splash screen.
- b. On the next panel, we chose option **3, Migration without saved variables**, since this was our first time using the customization dialog.
- c. On the next panel, we proceeded through each of the four options to allocate target data sets, define variables, generate migration jobs, and view the generated migration instructions.

Upon completion, the following data sets are created:

Data set name	Description
EBIT.WAS401.CNTL	Contains the generated migration jobs and instructions
EBIT.WAS401.DATA	Contains the IVP jobs, shell scripts, and other data

- d. We used option **S** to save the customized variables in a data set for future use. We specified the data set name as EBIT.WAS401.SAVECFG.
  - e. We then exited the dialog.
- 

4. We used FTP to copy the member containing the generated migration instructions, EBIT.WAS401.CNTL(BBOMISTR), to a workstation and then printed it.
- 

The remaining steps come from the generated migration instructions.

5. We ran the following jobs that were generated in the EBIT.WAS401.CNTL data set:
  - BBOMPAT2
  - BBOBIND
  - BBOIVPP
  - BBOIBN
  - BBOWMCP2
  - BBOMCFG

All jobs completed with a return code of 0.

---

6. We updated the following proclib members with the new high-level qualifier for the V4.0.1 product code data sets (SET BBOLIB='EBIT.WAS401'):

BBODMN  
BBONM  
BBONMS  
BBOSMS  
BBOSMSS  
BBOIR  
BBOIRS  
BBOASR1  
BBOASR1S  
BBOASR2  
BBOASR2S

---

7. We issued the following command to start the run-time server instances:

```
/S BBODMN.DAEMON01
```

---

8. Once all the servers were started, we started all of our application server instances (BBOASRxx). All servers started successfully and we saw message BBOU0579I for each server. The servers were now ready for the warm start.
- 

9. We issued the following command to start BBODMN in warm start mode:

```
/S BBODMN.DAEMON01,SRVNAME='DAEMON01',PARMS='-ORBCBI WARM'
```

All servers started successfully.

---

10. We then ran EBIT.WAS401.CNTL(BBOWCMIG). The job completed successfully.
- 

11. We then started all of our application servers in warm start mode. All servers started successfully.
- 

12. We successfully ran the V4.0.1 IVP jobs.
- 

This completes the migration to V4.0.1 on one system in our three-way WebSphere host cluster.

13. For each of the other two systems in our host cluster, we repeated the portion of step 1 on page 189 to update the LPA, followed by steps 7 through 12 above.

(In our sysplex, we use shared system data sets—linklist, parmlib, proclib, and so on—so we did not have to repeat the changes to them (for instance, APF authorizations and updating values in procs) on each system. See our

## WebSphere Application Server

related discussion earlier under “Enabling WebSphere Application Server V4.0 for z/OS and OS/390 on a sysplex” on page 181.)

---

This completes our migration from WebSphere for z/OS V4.0 to V4.0.1 on our sysplex.

## Additional migration experiences with WebSphere for z/OS V4.0.1

Although our testing of WebSphere for z/OS V4.0.1 is not yet complete, we experienced a few problems during our migration and early testing that we'd like to point out:

- Workload management (WLM) module IWMDNREG in z/OS V1R3 contained a bug that prevented our servers from coming up. APAR OW52213 (PTF UW84914) resolves this problem.
- Job BBOVCMIG failed when we tried to run it. The problem was that the wcmigrate.sh shell script was incorrectly passing the \$systemname variable to the wcmigrate.rexx exec. We updated wcmigrate.sh by putting double quotes around the \$systemname variable (" \$systemname"). The BBOVCMIG job then ran successfully. APAR PQ54282 (PTF UQ59243) resolves this problem.
- Job BBOIVPE failed the first time we ran it. We found that the JNDI name for the policy session was incorrect. We updated the SESSIONNAME variable in the ejbivp.sh shell script, as follows:

```
SESSIONNAME=/UTCPLXJ8/BBOASR2/PolicyIVP/policysession_deploy/ivp.policysession/com.ibm.ws390.samples.ivp.ejb.PolicySessionHome
```

The BBOIVPE job then ran successfully. APAR PQ55678 is open to address this problem.

- APAR PQ55449 (PTF UQ61650) resolves a potential system abend (S978, RC=4) in the daemon (BBODMN).
- We experienced a security problem that occurs when the Web server switches IDs using the WebSphere for z/OS V4.0.1 plugin. APAR PQ57800 is open to address this problem.
- We experienced a problem running our workload using multi clients. APARs OW50643 and OW53425 are open to address this problem.

---

## 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 HTTP Server for OS/390 documentation, available at <http://www.ibm.com/software/websphere/httpservers/library.html>
- IBM WebSphere Application Server for z/OS and OS/390 documentation, available at [http://www.ibm.com/software/webservers/appserv/library\\_390.html](http://www.ibm.com/software/webservers/appserv/library_390.html)
- IBM WebSphere Studio documentation, available at <http://www.ibm.com/software/websphere/studio/library.html>
- *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/>

---

## Chapter 15. Running Linux on IBM mainframe servers

This chapter discusses our experiences with setting up and running Linux on IBM mainframe servers.

Linux can run on IBM mainframe servers in any of the following configurations:

- in basic mode, running a single, native Linux image
- in LPAR mode, with one or more logical partitions each running a single Linux image
- one or more Linux images running as guests under IBM z/VM (with z/VM itself running in basic mode or LPAR mode)

This chapter covers the following topics:

- “Setting up Linux guests under z/VM” describes how we set up Linux guests under z/VM on our z900 server.
- “Setting up HiperSockets communications for LVMs” on page 200 describes how we set up HiperSockets communications for our Linux guests under z/VM.
- “Setting up LPAR CPU management for non-z/OS partitions” on page 205 describes how we enabled dynamic CPU management for our z/VM partition.

For additional information, see the following:

- Linux at IBM Web site at [www.ibm.com/linux/](http://www.ibm.com/linux/)
- Linux for zSeries Web site at [www.ibm.com/servers/eserver/zseries/os/linux/](http://www.ibm.com/servers/eserver/zseries/os/linux/)
- IBM Redbook *Linux for S/390*, SG24-4987, available at [www.ibm.com/redbooks](http://www.ibm.com/redbooks)

---

### Setting up Linux guests under z/VM

For our Linux setup, we followed the procedure described in the IBM Redpaper *Building Linux Systems under IBM VM*, which is available on the IBM Redbooks Web site at [www.ibm.com/redbooks/](http://www.ibm.com/redbooks/). This book describes one approach to building multiple Linux systems as Virtual Machine guests running under the IBM VM/ESA and z/VM operating systems (which we simply refer to as VM). We recommend that you get a copy of this Redpaper and read it along with the information in this section of our test report.

The following are the major tasks we performed to set up Linux guests under VM. The sections that follow provide additional information about each task. However, while we do not reiterate every step in each task (since they are already described in *Building Linux Systems under IBM VM*), we do discuss how aspects of each task specifically relate to our environment.

1. Preparing the DASD
2. Creating a VM user definition for the first Linux guest
3. Setting up IP network connectivity
4. Building the first Linux instance
5. Creating and tailoring additional Linux instances

For the purpose of reporting on our test experiences, we assume the existence of a base VM system and basic familiarity with VM system administration (including TCP/IP network administration). See the following publications for more information about these topics:

- *VM/ESA Planning and Administration*, SC24-5750, or *z/VM Planning and Administration*, SC24-5948

## Linux on IBM mainframe servers

- *VM/ESA CP Command and Utility Reference, SC24-5773, or z/VM CP Command and Utility Reference, SC24-5967*
- *VM/ESA TCP/IP Function Level 320 Planning and Customization, SC24-5847, or z/VM TCP/IP Level 3A0 Planning and Customization, SC24-5981*

The z/OS Internet library at [www.ibm.com/servers/eserver/zseries/zos/bkserv/](http://www.ibm.com/servers/eserver/zseries/zos/bkserv/) contains a link to the VM publications.

## Preparing the DASD

*Building Linux Systems under IBM VM* describes how to format and label the DASD volumes for Linux. These volumes will be used to allocate VM minidisks for swap space and a root file system for each Linux virtual machine, and a /usr file system to be shared by all Linux guests. Each Linux guest also requires a CMS-formatted minidisk as its A (191) disk to hold the PROFILE EXEC (and, for the first guest, the files that you upload to IPL and build the first Linux instance).

We followed the instructions to prepare our DASD. We're using a relatively small VM system with six 3390 DASD volumes. In our environment, we reserved DASD devices 2F3A through 2F3F for VM. We're currently using the volumes as follows:

Device	Volume	Purpose
2F3A	VE9IPL	VM SYSRES volume
2F3B	VMPK01	local CMS user minidisk allocations
2F3C	VMPK02	<b>Linux 191, 200, and 201 minidisk allocations</b>
2F3D	VMPK03	<b>Linux 202 minidisk allocation</b>
2F3E	VMPK04	future growth
2F3F	VMPK05	future growth

These 3390 volumes reside on an IBM Enterprise Storage Server.

## Creating a VM user definition for the first Linux guest

The VM user directory file, normally called USER DIRECT, contains the definitions for each VM user ID. VM considers each Linux guest as just another user of the system; it does not know or care that each of these guest virtual machines runs a whole new operating system environment. (Of course, from a human being's perspective, we think of each Linux guest as a separate "virtual server" running Linux.)

*Building Linux Systems under IBM VM* describes how to create Linux guests by manually editing the USER DIRECT file. You can follow this approach or, if your installation already has procedures in place for administering VM users (such as using the Directory Maintenance (DIRMAINT) program), you can follow your own local procedures.

We used the manual approach to edit the user directory to add the new Linux guest. Since we use RACF in our environment, we also used the RACF dialogs to define this new user to RACF.

Figure 20 on page 195 shows an example of the VM user directory entry that we created for our first Linux guest, which we named PETLNX01.

```

USER PETLNX01 password 992M 992M G
INCLUDE LINDFLT
IPL CMS PARM AUTOOCR
MACHINE ESA
SPECIAL 2000 CTCA TCPIP
SPECIAL 2001 CTCA TCPIP
MDISK 191 3390 0010 0030 VMPK02 MR ALL writepw multipw
MDISK 200 3390 0040 0200 VMPK02 MR ALL writepw multipw
MDISK 201 3390 1000 2000 VMPK02 MR ALL writepw multipw
MDISK 202 3390 0010 3000 VMPK03 MR ALL writepw multipw

```

Figure 20. Sample VM user directory entry for our first Linux guest

Note the following details about the VM directory entry shown in Figure 20.

- 1** The user name (that is, the VM user ID) of our first Linux guest is PETLNX01. We set the initial and maximum storage size to 992M.
- 2** We also defined the LINDFLT profile in the user directory so that we could include it within the directory entries for each of our Linux guests. The LINDFLT profile, shown in *Building Linux Systems under IBM VM*, contains directory statements that define certain virtual machine characteristics that will be the same for each guest. These include the virtual spooled unit record devices (reader, punch, and printer) and console, and links to certain system minidisks, such as those that contain the CMS system and other commands and utilities.
- 3** These lines define the virtual channel-to-channel adapter (CTCA) devices 2000 and 2001 between our Linux virtual machine and the TCP/IP virtual machine.
- 4** The MDISK statements define minidisks for the Linux guest, as follows:

Minidisk virtual address	DASD type	Size (in cylinders)	Real DASD volume	Purpose
191	3390	30	VMPK02	Guest "home" disk (holds PROFILE EXEC and Linux boot files)
200	3390	200	VMPK02	Linux swap space
201	3390	2000	VMPK02	/ (root file system, one per Linux instance)
202	3390	3000	VMPK03	/usr (shared among all Linux guests)

## Setting up IP network connectivity

There are several ways to connect a Linux guest to an IP network. For the configuration we're testing, we use virtual CTCs to communicate between Linux images, and between the Linux images and the VM TCP/IP stack. Virtual CTCs operate at memory speed and provide high bandwidth for TCP/IP traffic. Connectivity to other network hosts external to VM occurs through an OSA-2 feature which is defined to and driven by the VM TCP/IP stack. The use of the VM TCP/IP stack allows for centralized control of dynamic IP routing across all Linux guests.

Figure 21 on page 196 shows a high-level illustration of our Linux networking connectivity.

## Linux on IBM mainframe servers

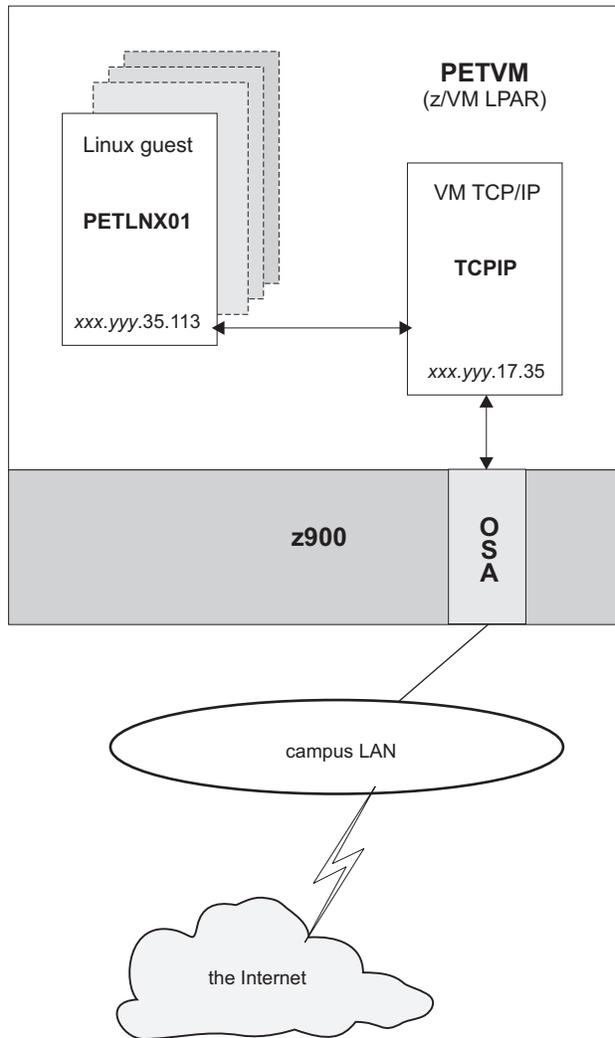


Figure 21. Overview of our Linux IP networking connectivity

To accomplish the connectivity shown in Figure 21, we updated the networking definitions in the following places:

### 1. VM user directory entry for our first Linux guest

As shown earlier in Figure 20 on page 195, the following statements define the virtual CTC devices that our PETLNX01 guest will use to communicate with the VM TCP/IP stack:

```
SPECIAL 2000 CTCA TCPIP
SPECIAL 2001 CTCA TCPIP
```

Since each VM user has its own virtual environment, each subsequent Linux guest that we create would also use virtual addresses 2000 and 2001 for its virtual CTC devices. Eventually, we will couple the Linux guest's virtual CTCs with a corresponding pair of virtual CTC devices that we define to the VM TCP/IP virtual machine.

### 2. VM user directory entry for VM TCP/IP

On our VM system, the name of the TCP/IP virtual machine is TCPIP. Figure 22 on page 197 shows a sample user directory entry for our TCP/IP virtual machine.

```

USER TCPIP password 32M 128M ABG
INCLUDE TCPCMSU
OPTION QUICKDSP SVMSTAT MAXCONN 1024 DIAG98 APPLMON
SHARE RELATIVE 3000
IUCV ALLOW
IUCV ANY PRIORITY
IUCV *CCS PRIORITY MSGLIMIT 255
LINK TCPMAINT 591 591 RR
LINK TCPMAINT 592 592 RR
LINK TCPMAINT 198 198 RR
MDISK 191 3390 3005 5 +VMRES MR readpw writepw multipw
SPECIAL 2000 CTCA PETLNX01
SPECIAL 2001 CTCA PETLNX01

```

Figure 22. Sample VM user directory entry for TCP/IP

The user directory entry for the TCP/IP virtual machine contains the following statements to define a pair of virtual CTC devices to communicate with our PETLNX01 Linux guest:

```

SPECIAL 2000 CTCA PETLNX01
SPECIAL 2001 CTCA PETLNX01

```

**Note:** We happen to be using the same virtual addresses (2000 and 2001) to define this pair of CTCs to our TCPIP virtual machine. However, the fact that these device addresses are the same as the ones that we defined to our PETLNX01 virtual machine is purely coincidental. As noted above, while additional Linux guests will *each* use addresses 2000 and 2001 for their CTC devices, we would need to define additional even/odd pairs of addresses (for instance, 2002 and 2003, 2004 and 2005, and so on) for the corresponding CTC devices in VM TCP/IP.

### 3. PROFILE TCPIP for the VM TCP/IP stack

We updated our VM TCP/IP profile, PROFILE TCPIP, to add the network interface definitions for our first Linux instance. Figure 23 on page 198 shows the relevant portions of our PROFILE TCPIP to illustrate the configuration of our token ring OSA and the new CTC connection to our first Linux instance.

**Note:** Note that we run ROUTED, but we also show (in comments) the tailored GATEWAY statement that would otherwise apply.

## Linux on IBM mainframe servers

```
:
:
; Link to token ring OSA
DEVICE DEVTR1 LCS 6D0
LINK LNKTR1 IBMTR 0 DEVTR1

; Virtual CTC to PETLNX01
DEVICE CTCDVLN1 CTC 2000
LINK CTCLNX1 CTC 0 CTCDVLN1
:
:
HOME
xxx.yyy.17.35 LNKTR1
xxx.yyy.35.113 CTCLNX1
:
:
; Routing information (if not using the ROUTED server)
; GATEWAY
; Network First hop Driver Packet size Subn mask Subn value
; xxx = LNKTR1 1492 0.255.255.0 0.yyy.17.0
; xxx.yyy.17.113 = CTCLNX1 1492 HOST
; DEFAULTNET xxx.yyy.17.1 LNKTR1 1492 0
;
;
;
; If you are using ROUTED, comment out the GATEWAY statement and
; update the BSDROUTINGPARMS statement to reflect your installation
; configuration and remove the semicolon
;
; Link Maxmtu Metric Subnet Mask Dest Addr
;
; BSDROUTINGPARMS false
; LNKTR1 1492 0 255.255.255.0 xxx.yyy.17.1
; CTCLNX1 65535 1 255.0.0.0 xxx.yyy.17.113
; ENDBSDROUTINGPARMS
:
:
START DEVTR1
START CTCDVLN1
```

Figure 23. Portions of our PROFILE TCPIP on VM showing our networking interface definitions for our first Linux guest

#### 4. PROFILE EXEC for our Linux guest

After you IPL CMS in a virtual machine, it automatically executes a file called PROFILE EXEC, if it exists. Once we logon to our PETLNX01 virtual machine for the first time (which we'll do in "Building the first Linux instance" on page 199), we'll create a PROFILE EXEC file to issue some commands to tailor our virtual environment for Linux. This includes connecting the virtual CTC devices between our Linux guest and the VM TCP/IP virtual machine.

To do this, we'll use the CP COUPLE command to couple the CTC read channel on the Linux guest to the write channel on VM TCP/IP, and couple the Linux write channel to the VM TCP/IP read channel:

```
CP COUPLE 2000 TCPIP 2001
CP COUPLE 2001 TCPIP 2000
```

See "Building the first Linux instance" on page 199 for an example of a complete PROFILE EXEC for the Linux guest.

## Building the first Linux instance

We followed the instructions in the IBM Redpaper *Building Linux Systems under IBM VM* to build our first Linux system image or instance. You can refer to that document for the detailed steps which we found worked very well. We'll simply highlight the major tasks below:

### 1. Logging on and setting up the virtual machine environment

The first time we logged on to the PETLNX01 virtual machine, we formatted our A (191) minidisk for CMS and then created the PROFILE EXEC. The following is an example of the contents of the PROFILE EXEC:

```
/* VM LINUX Profile exec */
/* trace ira */
'VMFCLEAR'
'CP SET MSG ON'
'CP SET EMSG ON'
'SET IMPCP ON'
'SET RUN ON'
'CP TERM CHARDEL OFF'
'VMLINK TCPMAINT 592 (NONAMES'
'CP SET PF10 RETRIEVE'
'CP SET RETR MAX'
'CP TERM HOLD OFF'
'CP TERM MORE 1 1'
'CP COUPLE 2000 TCPIP 2001'
'CP COUPLE 2001 TCPIP 2000'
```

Figure 24. Sample PROFILE EXEC for our Linux virtual machine

We then manually executed the profile.

Formatting the 191 minidisk and creating and manually running the profile are all one-time setup tasks. From now on, the profile will automatically execute when the PETLNX01 user logs on and IPLs CMS.

### 2. Uploading the Linux IPL material to VM

For this task, we used FTP to upload the Linux boot files from the workstation containing the SuSE starter system files to our A (191) minidisk. The Redpaper includes a transcript of a sample FTP session that demonstrates how to do this.

### 3. Booting Linux from the virtual reader

As the Redpaper describes, we created the LIN EXEC to easily boot the Linux system. The LIN EXEC punches the boot files from our A minidisk to our virtual reader, then IPLs the card images from the virtual reader to begin booting the Linux system.

We kept clearing the screen as it filled with messages and, after about 15 or 20 minutes, we eventually got to the Welcome text and the prompt to select the type of network device we were using. We selected **Channel to Channel**, then proceeded to reply to the prompts to specify our host name, IP addresses, MTU size, and temporary installation password.

Once our network setup was finished, we proceeded to build the first Linux instance.

### 4. Building the first Linux instance using YaST

We used telnet to login to our Linux system as **root** using the temporary password we had just set in the previous task.

**Note:** For this telnet session, you must use a terminal size of at least 80 x 25 lines to accommodate the YaST panels. (Some telnet packages default to

## Linux on IBM mainframe servers

only 24 lines.)

We followed the instructions to set up our disk devices and to run the YaST installation program, replying to the prompts as described in the documentation. (The YaST installation was very similar to our previous installation under S/390 Virtual Image Facility for Linux, described in our December 2001 edition.)

### 5. Rebooting the newly installed Linux system

After we completed the installation, we issued the Linux **halt** command to shut down the Linux system. Once the Linux system was halted, we IPLed CMS and prepared to reboot our newly installed Linux system from DASD.

We issued the CP command **IPL 201 CLEAR** to begin booting the Linux system from our 201 minidisk. Once the system finished booting, we were able to login again as **root**.

We can now use this Linux system as a master image for cloning additional Linux instances.

## Creating and tailoring additional Linux instances

Once the first Linux instance is complete, *Building Linux Systems under IBM VM* describes how to create and tailor additional Linux instances based on cloning the first Linux instance. We followed those instructions and successfully cloned several additional Linux instances.

---

## Setting up HiperSockets communications for LVMs

The zSeries HiperSockets function, also known as internal queued direct I/O (iQDIO), is a new zSeries microcode function that provides reliable, secure, high-performance TCP/IP communications between logical partitions within the same CPC without the need for any channel hardware (for instance, OSA cards and cabling). The result is a simulated virtual LAN within the server which operates at the speed of cross-memory operations. This is the fastest option for network communications between the LPs within a CPC. (For more information, see the IBM Redpaper *zSeries HiperSockets*, available at [www.redbooks.ibm.com/redpapers/pdfs/redp0160.pdf](http://www.redbooks.ibm.com/redpapers/pdfs/redp0160.pdf).)

Running under z/VM, you can use HiperSockets for communication among multiple guest images running Linux. To Linux, an iQDIO CHPID appears as an ordinary network interface.

**Note:** This section describes how to use “real” HiperSockets devices with Linux virtual machines. We call these real devices (even though they don’t use any channel hardware) because we define them in the IOCDs and can share them across LPs within the CPC. Contrast that with simply defining a virtual LAN under z/VM and defining virtual HiperSockets NIC devices in each Linux virtual machine. The latter method only provides for communications between guest images running under the same z/VM image. However, we can use the real HiperSockets devices to eventually connect our Linux images to our z/OS images running in other LPs on the same CPC.

Figure 25 on page 201 shows an overview of Linux networking using HiperSockets:

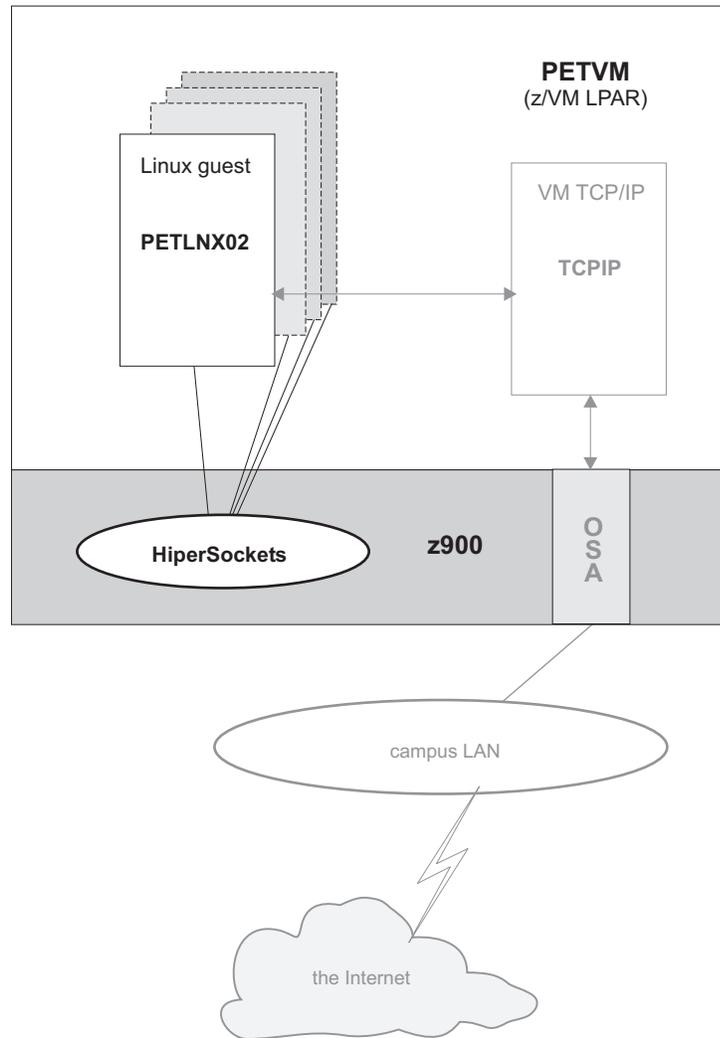


Figure 25. Overview of Linux networking with HiperSockets

Setting up HiperSockets communications for Linux guests under z/VM involves the following tasks:

- Defining the iQDIO hardware configuration
- Adding HiperSockets links to the Linux virtual machine configuration
- Configuring the network interface under Linux

The following sections describe how we carried out these tasks.

## Defining the iQDIO hardware configuration

We defined our new iQDIO channel paths, control units, and devices using the new channel type, IQD. (If you use HCD, APAR OW45976 adds support for the IQD channel type in HCD.) Note that the control units and devices defined to IQD channel paths do not represent real I/O devices but data in a TCP/IP stack.

You can define up to four IQD CHPIDs per CPC and you can share the CHPIDs among all LPs. For each IQD CHPID, you must also specify a maximum frame size. The valid values for the maximum frame size are: 16K (the default), 24K, 40K, and

## Linux on IBM mainframe servers

64K bytes, and you specify these as OS=00, OS=40, OS=80, and OS=C0, respectively. For our testing, we decided to specify four IQD CHPIDs, one for each of the four possible frame sizes.

The following is an example of our IQD hardware definitions:

```
⋮
CHPID PATH=(E0),SHARED, *
PARTITION=((JF0,J90,PETVM,Z0,Z1),(JF0,J90,PETVM,Z0,Z1)) *
TYPE=IQD
CHPID PATH=(E1),SHARED, *
PARTITION=((JF0,J90,PETVM,Z0,Z1),(JF0,J90,PETVM,Z0,Z1)) *
OS=40,TYPE=IQD
CHPID PATH=(E2),SHARED, *
PARTITION=((JF0,J90,PETVM,Z0,Z1),(JF0,J90,PETVM,Z0,Z1)) *
OS=80,TYPE=IQD
CHPID PATH=(E3),SHARED, *
PARTITION=((JF0,J90,PETVM,Z0,Z1),(JF0,J90,PETVM,Z0,Z1)) *
OS=C0,TYPE=IQD
⋮
CNTLUNIT CUNUMBR=E000,PATH=(E0),UNIT=IQD
CNTLUNIT CUNUMBR=E100,PATH=(E1),UNIT=IQD
CNTLUNIT CUNUMBR=E200,PATH=(E2),UNIT=IQD
CNTLUNIT CUNUMBR=E300,PATH=(E3),UNIT=IQD
⋮
IODEVICE ADDRESS=(E000,032),CUNUMBR=(E000),UNIT=IQD
IODEVICE ADDRESS=(E100,032),CUNUMBR=(E100),UNIT=IQD
IODEVICE ADDRESS=(E200,032),CUNUMBR=(E200),UNIT=IQD
IODEVICE ADDRESS=(E300,032),CUNUMBR=(E300),UNIT=IQD
⋮
```

The IQD frame size also determines the TCP/IP maximum transmission unit (MTU) size, as follows:

Table 16. Values for IQD frame sizes and resulting TCP/IP MTU sizes

IQD frame size (IOCP value)	TCP/IP MTU size
16K (default or OS=00)	8K
24K (OS=40)	16K
40K (OS=80)	32K
64K (OS=C0)	56K

For more information, see *z/OS HCD Planning, GA22-7525*, and *z/OS Communications Server: IP Configuration Guide, SC31-8775*. (Note that the appendix "Using HCD" in *z/OS Communications Server: IP Configuration Guide*, specifically at the SC31-8775-01 level, contains an example of using HCD to define IQD channels and devices.)

## Adding HiperSockets links to the Linux virtual machine configuration

After we activated the new IOCDs containing our HiperSockets definitions, the new device addresses were visible to the z/VM V4R2 system running on our PETVM LP.

**Note:** The fix for APAR VM62942 provides iQDIO support for z/VM V4R2 and earlier releases.

We then updated the VM directory entries for our Linux virtual machines and dedicated the HiperSockets devices to each Linux virtual machine. Figure 26 on page 203

page 203 shows an example of the DEDICATE statements for the devices that we added to our Linux VM directory entries:

```

USER PETLNX02 password 960M 1152M G
INCLUDE LINDFLT
IPL CMS PARM AUTOOCR
MACHINE ESA
SPECIAL 2000 CTCA PETLNX01
SPECIAL 2001 CTCA PETLNX01
SPECIAL 3000 CTCA PETLNX03
SPECIAL 3001 CTCA PETLNX03
DEDICATE 9A0 9A0
DEDICATE 9A1 9A1
DEDICATE 9A2 9A2
DEDICATE E000 E000
DEDICATE E001 E001
DEDICATE E002 E002
DEDICATE E003 E003
DEDICATE E300 E300
DEDICATE E301 E301
DEDICATE E302 E302
DEDICATE E303 E303
MDISK 191 3390 0010 0050 VMPK05 MR ALL writepw multipw
MDISK 201 3390 0060 2800 VMPK05 MR ALL writepw multipw
MDISK 251C 3390 0001 END VMPK0F MR ALL writepw multipw
MDISK 251D 3390 0001 END VMPK10 MR ALL writepw multipw
MDISK 251E 3390 0001 END VMPK11 MR ALL writepw multipw
MDISK 251F 3390 0001 END VMPK12 MR ALL writepw multipw

```

Figure 26. Sample Linux VM user directory entry containing HiperSockets devices

A HiperSockets interface requires three I/O devices—one for read control, one for write control, and one for data exchange—and the first device must have an even-numbered device address. (As the above example shows, we also dedicated the fourth device in the range (which is odd-numbered) to Linux, even though this device is not used for anything. Don't ask us why—we probably just didn't want it to feel left out.)

We decided to test with the smallest (16K) and largest (64K) IQD frame sizes. As shown above, we dedicated two sets of IQD devices to Linux: E000-E003 (which use the default 16K bytes frame size) and E300-E303 (which use the 64K bytes frame size).

## Configuring the network interface under Linux

Once our Linux virtual machines had access to the IQD devices, we configured the network adapters under Linux. We configured two network interfaces—one using the IQD devices with the 16K bytes frame size and one using the devices with the 64K bytes frame size.

We're using SuSE Linux Enterprise Server 7 for S/390 with kernel 2.4.7 which contains support for HiperSockets. (You can visit [www.suse.com/](http://www.suse.com/) to learn more about SuSE Linux.)

### Steps to configure the network interfaces

We did the following to configure the network interfaces under Linux:

1. We updated the `/etc/chandev.conf` file and added the highlighted lines as shown below. You can also use YaST to do this.

## Linux on IBM mainframe servers

```
noauto;qeth2,0x09a0,0x09a1,0x09a2;add_parms,0x10,portname:PETLINUX
add_model,0x10,0x1731,5,0x1732,5,0,0
qeth0,0xe000,0xe001,0xe002,0,0
qeth1,0xe300,0xe301,0xe302,0,0
ctc0,0x3000,0x3001,0,0
```

Notice that we only configure three devices for each interface:

- Interface qeth0 uses devices e000, e001, and e002. This interface operates with an IQD frame size of 16K bytes (as specified by the hardware configuration above) and, thereby, a TCP/IP MTU size of 8K bytes.
- Interface qeth1 uses devices e300, e301, and e302. This interface operates with an IQD frame size of 64K bytes and, thereby, a TCP/IP MTU size of 56K bytes.

- 
2. We updated the /etc/Modules.conf file and added the highlighted lines as shown below:

```
Aliases - specify your hardware
alias hsi0 qeth
alias hsi1 qeth
alias eth2 qeth
#alias eth0 off
```

- 
3. We re-booted the Linux image to pick up the hardware configuration changes.

- 
4. We configured the interfaces with the appropriate IP addressing parameters:

```
ifconfig hsi0 9.xxx.yyy.aaa netmask 255.255.255.0 broadcast 9.xxx.yyy.bbb
```

```
hsi0 Link encap:Ethernet HWaddr 00:00:00:00:00:00
 inet addr:9.xxx.yyy.aaa Mask:255.255.255.0
 inet6 addr: fe80::200:ff:fe00:0/10 Scope:Link
 UP RUNNING NOARP MULTICAST MTU:8192 Metric:1
 RX packets:0 errors:0 dropped:0 overruns:0 frame:0
 TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:100
 RX bytes:0 (0.0b) TX bytes:264 (264.0b)
 Interrupt:5
```

```
ifconfig hsi1 9.xxx.yyy.ccc netmask 255.255.255.0 broadcast 9.xxx.yyy.bbb
```

```
hsi1 Link encap:Ethernet HWaddr 00:00:00:00:00:00
 inet addr:9.xxx.yyy.ccc Mask:255.255.255.0
 inet6 addr: fe80::200:ff:fe00:0/10 Scope:Link
 UP RUNNING NOARP MULTICAST MTU:57344 Metric:1
 RX packets:0 errors:0 dropped:0 overruns:0 frame:0
 TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
 collisions:0 txqueuelen:100
 RX bytes:0 (0.0b) TX bytes:264 (264.0b)
 Interrupt:5
```

### Notes:

- a. As with any IP networking, HiperSockets networking is fully routable. However, since we place all of our Linux images on the same subnet, we have no need to define a router (or gateway) address.
- b. The hardware addresses above are composed of all zeros—this is because HiperSockets uses no physical hardware devices.

- c. The MTU size for hsi0 (which uses the 16K IQD frame size) is 8192 (or 8K) bytes and the MTU size for hsi1 (which uses the 64K IQD frame size) is 57344 (or 56K) bytes. Linux automatically sets these MTU sizes based on the hardware frame sizes that it detects.

---

The HiperSockets interfaces are now available for network communications.

## Experiences with HiperSockets communications

We used an FTP workload to exercise the HiperSockets connections between our Linux guest images. This workload simply uses Linux-to-Linux FTP sessions to transfer a variety of files of different sizes. Although we did not perform any formal performance testing, we observed a marked increase in speed when running this workload over HiperSockets connections as compared to running over virtual CTCs.

---

## Setting up LPAR CPU management for non-z/OS partitions

Starting with z/OS V1R2, you can extend the LPAR CPU management function of the Intelligent Resource Director (see the discussion of IRD in our December 2001 edition) to include management of non-z/OS partitions. LPAR CPU management for non-z/OS partitions allows WLM to shift weight from either z/OS or non-z/OS partitions to another z/OS or non-z/OS partition within the same LPAR cluster.

**Note:** This function is shipped disabled in z/OS V1R2. You must apply the fix for APAR OW50221 to enable this function for use.

For additional information, see *z/OS MVS Planning: Workload Management*, SA22-7602.

This section describes how we set up non-z/OS partition CPU management for the z/VM partition that supports our virtual Linux images.

## Steps for setting up LPAR CPU management for our z/VM partition

We did the following to set up non-z/OS partition CPU management for our z/VM partition:

**Before we began:** We had previously enabled LPAR CPU management for our z/OS partitions (see our December 2001 edition), so we had already performed the set-up work, such as:

- Defining the WLM structure (SYSZWLM\_XXXXYYYY)
- Setting the z/OS partitions to be WLM-managed
- Setting the z/OS partition weights
- Making sure the partitions are set to use shared CPUs

We also made sure we installed the fixes for APARs OW49956, OW50221, and OW51586.

We continued with the following steps:

1. On the HMC (Primary Support Element Workplace), we updated the IMAGE Profile for our z/VM partition, as follows: (Note that the system name for our z/VM system is PETVM.)
  - Set the partition weight to 10
  - Set the number of processors to 2

## Linux on IBM mainframe servers

- Set the CP Management cluster name to UCTPLXJ8, which matches our sysplex name
- Selected the NOT DEDICATED PROCESSORS option
- Selected the ENABLED WORKLOAD MANAGER option

These changes take effect when the partition is re-activated.

- 
2. In the WLM application, we updated the service definitions by creating a new LINUX service class with velocity goal, as follows:

```
* Service Class LINUX - Linux CPU Management

Created by user G177187 on 2001/09/04 at 13:11:19
Base last updated by user G177187 on 2001/09/04 at 13:11:19

Base goal:
CPU Critical flag: NO

Duration Imp Goal description
- - - - -
1 3 Execution velocity of 70
```

- 
3. We created a SYSH subsystem with no default service class and defined a classification rule with the SY qualifier. The SY qualifier specifies the system name of our z/VM system (PETVM) and the LINUX service class that we created in the previous step. The definition looks like this:

```
There is no default service class.
There is no default report class.

Qualifier Qualifier Starting Service Report
type name position Class Class
- - - - -
1 SY PETVM LINUX
```

---

This completes the setup of LPAR CPU management for our z/VM partition.

So far, we have only set up this new function but have little experience with it in operation. As we gain more experience with this function, we hope to report on it in a future test report.

## 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 V1R3 and z/OS.e V1R3. 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 V1R3 and z/OS.e V1R3

Member name	z/OS release	Change summary	Related to
IFAPRDxx	z/OS.e V1R3	<ul style="list-style-type: none"> <li>Created a separate IFAPRD02 member for z/OS.e</li> <li>Changed product IDs to ID(5655-652)</li> <li>Removed entries for priced features not supported in z/OS.e V1R3: <ul style="list-style-type: none"> <li>BDT File-to-File</li> <li>BookManager BUILD</li> <li>GDDM REXX</li> <li>GDDM PGF</li> </ul> </li> </ul>	z/OS.e, dynamic enablement
IEASYMPT	z/OS.e V1R3	Made the following changes only for our z/OS.e image (JH0): <ul style="list-style-type: none"> <li>Changed the LPARNAME() value to Z0SEJH0</li> <li>Changed the value of the &amp;PROD symbol definition to point to our new IFAPRDxx member</li> </ul>	z/OS.e, dynamic enablement
IEASYSxx	z/OS.e V1R3	Created a separate IEASYSxx member for z/OS.e and added LICENSE=Z/0SE statement	z/OS.e
LOADxx	z/OS V1R3	Changed the PARMLIB statement to use SYS1.PETR13.PARMLIB	concatenated parmlib
(Actually in SYS0.IPLPARM. See note below.)	z/OS.e V1R3	Made the following changes only for our z/OS.e image (JH0): <ul style="list-style-type: none"> <li>Changed the LPARNAME value to Z0SEJH0</li> <li>Changed the SYSPARM statement to point to our separate IEASYSxx member</li> </ul>	z/OS.e
LPALSTxx	z/OS V1R3	Added: SYS1.SDWDDLPA	DFSMS (CICSVR support)
		Added: SYS1.SIATLPA (Functionally equivalent to and unchanged from z/OS V1R2 JES3)	JES3
PROGxx	z/OS V1R3	Added: SYS1.SHASLINK SYS1.SHASMIG (Functionally equivalent to and unchanged from z/OS V1R2 JES2)	JES2
(APF additions)			
PROGyy	z/OS V1R3	Added to LNKLSTxx: SYS1.SHASLINK SYS1.SHASMIG (Functionally equivalent to and unchanged from z/OS V1R2 JES2)	JES2
(LNKLST)			
		Added to LNKLSTxx: SYS1.SIATLIB SYS1.SIATLINK SYS1.SIATMIG (Functionally equivalent to and unchanged from z/OS V1R2 JES3)	JES3

## Parmlib members

Table 17. Summary of our parmlib changes for z/OS V1R3 and z/OS.e V1R3 (continued)

Member name	z/OS release	Change summary	Related to
<b>Note:</b> 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.			

## 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 29.

### 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 29 on page 211.

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 27. 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 29 on page 211, 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 1 83 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 28. 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.

WORKLOAD ACTIVITY

OS/390 SYSPLEX UTCPLXJ8 DATE 01/18/2001 INTERVAL 30.00.307 MODE = GOAL
REL. 02.10.00 RPT VERSION 02.10.00 TIME 10.00.00

POLICY ACTIVATION DATE/TIME 12/07/2000 09.50.19

REPORT BY: POLICY=WLMPOL01 WORKLOAD=CICS SERVICE CLASS=CICS RESOURCE GROUP=\*NONE PERIOD=1 IMPORTANCE=2
CRITICAL =NONE

TRANSACTIONS TRANS.-TIME HHH.MM.SS.TTT
AVG 0.00 ACTUAL 54
MPL 0.00 EXECUTION 81
ENDED 1733071 QUEUED 0
END/S 962.79 R/S AFFINITY 0
#SWAPS 0 INELIGIBLE 0
EXCTD 1350758 CONVERSION 0
AVG ENC 0.00 STD DEV 255
REM ENC 0.00
MS ENC 0.00

Table with columns: SUB TYPE, P, TOTAL, ACTIVE, READY, IDLE, CONV, I/O, PROD, MISC, LOCK, WAITING FOR, STATE, SWITCHED TIME (%), LOCAL, SYSPL, REMOT. Rows include CICS BTE, CICS EXE, DB2 EXE, SMS EXE.

VELOCITY MIGRATION: I/O MGMT N/A INIT MGMT N/A

Table with columns: ---RESPONSE TIME---, EX, PERF, HH.MM.SS.TTT, VEL, INDX, Comman. Rows include GOAL, ACTUALS, \*ALL, JA0, JB0, JC0, JE0, JF0, JG0, JH0, JB0, J90, Z0.

OS/390 SYSPLEX UTCPLXJ8 DATE 01/18/2001 INTERVAL 30.00.307 MODE = GOAL
REL. 02.10.00 RPT VERSION 02.10.00 TIME 10.00.00

POLICY ACTIVATION DATE/TIME 12/07/2000 09.50.19

Table with columns: ---TIME---, --NUMBER OF TRANSACTIONS--, -----PERCENT-----, CUM TOTAL, IN BUCKET. Rows show time intervals from <= 00.00.00.300 to > 00.00.02.400.

REPORT BY: POLICY=WLMPOL01 WORKLOAD=CICS
cics workload

Table with columns: TRANSACTIONS, TRANS.-TIME, HHH.MM.SS.TTT, --DASD I/O--, ---SERVICE---, --SERVICE RATES--, PAGE-IN RATES, ----STORAGE----. Rows include AVG, MPL, ENDED, END/S, #SWAPS, EXCTD, AVG ENC, REM ENC, MS ENC.

WORKLOAD ACTIVITY

Figure 29. Example RMF workload activity report in WLM goal mode



---

## Appendix C. Document availability and distribution

The following information describes the variety of ways in which you can obtain our test reports.

**Availability on the Internet:** You can view, download, and print the most current quarterly edition of our test report from our z/OS Integration Test Web site at:

[www.ibm.com/servers/eserver/zseries/zos/integtst/](http://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 quarterly 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/](http://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 sections. 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/](http://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](http://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).

### 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 will no longer produce a hardcopy edition of our year-end test reports. You can still order a printed copy of a previous year-end

## Document distribution

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>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*.

**Availability on the IBM intranet:** IBM customer account representatives can also access all editions and formats of our test report through the Server Sales intranet site, either for their own use or to give to customers. The intranet site links to the same documents that are available on our external Web site.

---

## Appendix D. 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> )	<a href="http://www.ibm.com/ibm/terminology/">www.ibm.com/ibm/terminology/</a>
<i>IBM HTTP Server library</i>	<a href="http://www.ibm.com/software/websphere/httpservers/library.html">www.ibm.com/software/websphere/httpservers/library.html</a>
<i>IBMLink</i>	<a href="http://www.ibm.com/ibmlink/">www.ibm.com/ibmlink/</a>
<i>IBM mainframe servers Internet library</i>	<a href="http://www.ibm.com/servers/eserver/zseries/library/">www.ibm.com/servers/eserver/zseries/library/</a>
<i>IBM Redbooks</i>	<a href="http://www.ibm.com/redbooks/">www.ibm.com/redbooks/</a>
<i>IBM Systems Center Publications</i>	<a href="http://www.ibm.com/support/techdocs/">www.ibm.com/support/techdocs/</a>
<i>Linux at IBM</i>	<a href="http://www.ibm.com/linux/">www.ibm.com/linux/</a>
<i>Net.Data Library</i>	<a href="http://www.ibm.com/software/data/net.data/library.html">www.ibm.com/software/data/net.data/library.html</a>
<i>OS/390 e-business Integration Test (ebIT)</i>	<a href="http://www.ibm.com/servers/eserver/zseries/ebusiness/techinfo/ebitintr.html">www.ibm.com/servers/eserver/zseries/ebusiness/techinfo/ebitintr.html</a>
<i>OS/390 Internet library</i>	<a href="http://www.ibm.com/servers/s390/os390/bkserv/">www.ibm.com/servers/s390/os390/bkserv/</a>
<i>Parallel Sysplex</i>	<a href="http://www.ibm.com/servers/eserver/zseries/psa/">www.ibm.com/servers/eserver/zseries/psa/</a>
<i>Parallel Sysplex Customization Wizard</i>	<a href="http://www.ibm.com/servers/eserver/zseries/zos/wizards/parallel/">www.ibm.com/servers/eserver/zseries/zos/wizards/parallel/</a>
<i>System Automation for OS/390</i>	<a href="http://www.ibm.com/servers/eserver/zseries/software/sa/">www.ibm.com/servers/eserver/zseries/software/sa/</a>
<i>WebSphere Application Server</i>	<a href="http://www.ibm.com/software/webervers/appserv/">www.ibm.com/software/webervers/appserv/</a>
<i>WebSphere Application Server library</i>	<a href="http://www.ibm.com/software/webervers/appserv/library_390.html">www.ibm.com/software/webervers/appserv/library_390.html</a>
<i>WebSphere Studio library</i>	<a href="http://www.ibm.com/software/websphere/studio/library.html">www.ibm.com/software/websphere/studio/library.html</a>
<i>z/OS Consolidated Service Test</i>	<a href="http://www.ibm.com/servers/eserver/zseries/zos/servicetst">www.ibm.com/servers/eserver/zseries/zos/servicetst</a>
<i>z/OS Integration Test</i> (includes information from OS/390 Integration Test)	<a href="http://www.ibm.com/servers/eserver/zseries/zos/integtst/">www.ibm.com/servers/eserver/zseries/zos/integtst/</a>
<i>z/OS Internet library</i>	<a href="http://www.ibm.com/servers/eserver/zseries/zos/bkserv/">www.ibm.com/servers/eserver/zseries/zos/bkserv/</a>
<i>z/OS UNIX System Services</i>	<a href="http://www.ibm.com/servers/eserver/zseries/zos/unix/">www.ibm.com/servers/eserver/zseries/zos/unix/</a>
<i>z/OS.e home page</i>	<a href="http://www.ibm.com/servers/eserver/zseries/zose/">www.ibm.com/servers/eserver/zseries/zose/</a>
<i>z/OS.e Internet library</i>	<a href="http://www.ibm.com/servers/eserver/zseries/zose/bkserv/">www.ibm.com/servers/eserver/zseries/zose/bkserv/</a>

---

## 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*

<b>Web site name or topic</b>	<b>Web site address</b>
<i>Cisco Systems</i>	<a href="http://www.cisco.com/">www.cisco.com/</a>
<i>Java Servlet Technology</i>	<a href="http://java.sun.com/products/servlet/">java.sun.com/products/servlet/</a>
<i>Java 2 Platform, Enterprise Edition (J2EE)</i>	<a href="http://java.sun.com/products/j2ee/">java.sun.com/products/j2ee/</a>
<i>JavaServer Pages (JSP)</i>	<a href="http://java.sun.com/products/jsp/">java.sun.com/products/jsp/</a>
<i>SuSE Linux</i>	<a href="http://www.suse.com/">www.suse.com/</a>

---

## Appendix E. 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 it 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 Volume I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.



---

## 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	MVS/ESA
BatchPipes	Net.Data
BookManager	NetView
BookMaster	OS/2
CICS	OS/390
CICSplex	Parallel Sysplex
DB2	QMF
DB2 Connect	RACF
DFS	RAMAC
DFSMS/MVS	Redbooks
DFSMSHsm	RETAIN
DFSMSrmm	RMF
Encina	RS/6000
Enterprise Storage Server	S/390
ESCON	S/390 Parallel Enterprise Server
@server	SecureWay
FICON	SP
GDDM	Sysplex Timer
Hiperbatch	Tivoli
IBM	VisualAge
IBM.COM	VM/ESA
IBMLink	VTAM
IMS	WebSphere
Infoprint	z/Architecture
Language Environment	z/OS
Library Reader	z/OS.e
MQSeries	z/VM
MVS	zSeries

Domino is a trademark of International Business Machines Corporation and Lotus Development Corporation in the United States, other countries, or both.

The following terms are trademarks of other companies:

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

Microsoft, Windows, and Windows NT are registered 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 217
- application enablement
  - configuration 57
  - workloads 65
- ATM LAN emulation
  - configuration 59
- automation
  - See *also* Parallel Sysplex automation
  - Parallel Sysplex 51
- availability
  - of this document 213

## C

- channel connectivity
  - coupling facility channels 10
  - ESCON channels 11
  - FICON channels 11
- chtag command 154
- CICS TS 2.2
  - migrating to 45
  - migration experiences 49
  - overview of migration 45
  - performing the migration 46
    - migrating CICSPlex SM 47
    - migrating the CASs 47
    - migrating the CMASs 48
    - migrating the MASs 49
    - preparing for migration 46
- common recall queue
  - See DFSMSHsm
- configuration
  - application enablement 57
  - ATM LAN emulation 59
  - coupling facility channels 10
  - Ethernet LAN 58
  - hardware details 5
    - mainframe servers 5
  - hardware overview 3
  - LDAP Server overview 125
  - networking 57
  - Parallel Sysplex hardware 3
  - sysplex hardware details
    - coupling facilities 7
    - other sysplex hardware 10
  - sysplex software 12
  - token-ring LAN 59
  - VTAM 14
- customization dialog, WebSphere for z/OS 190

## D

- DFSMSHsm
  - common recall queue (CRQ) 43
    - displaying and controlling CRQ processing 44
    - experiences 44

- DFSMSHsm (*continued*)
  - common recall queue (CRQ) (*continued*)
    - starting CRQ processing 44
    - steps for defining 43
- disability 217
- distribution
  - of this document 213
- dynamic enablement
  - relation to IBM License Manager 12
  - relation to IFAPRDxx parmlib member 12

## E

- Enhanced ASCII support 143
  - automatic conversion 143
  - controlling 147
    - c option of mount shell command 150
    - AUTOCVT in BPXPRMxx 147
    - C/C++ compiler and run-time support 148
  - chtag command 154
  - environment variables, using 152
  - FILETAG(,AUTOTAG) run-time option 149
  - FILETAG(AUTOCVT) run-time option 149
  - LE FILETAG() run-time option 148
  - mount panel under TSO ISHELL 151
  - redirection using shell variables 153
  - setting up automount 151
  - TAG() parameter of TSO MOUNT command 150
- displaying file tags 146
- file tagging 143
- implementing 145
- overview 143
- environment
  - networking and application enablement 57
  - Parallel Sysplex 3
  - workloads 15
- environment variables, using to control Enhanced ASCII support 152
- ESCON channels 11
- Ethernet
  - 10BASE-T 58
  - Fast Ethernet 58
  - Gigabit Ethernet 58
- Ethernet LAN
  - configuration 58

## F

- FICON channels 11
  - FICON native (FC) mode 11

## H

- hardware
  - configuration details 5
    - mainframe servers 5
  - configuration overview 3

hardware (*continued*)  
  coupling facility channel configuration 10  
  Parallel Sysplex configuration 3  
HiperSockets  
  defining the hardware configuration 201  
  setting up Linux guests under z/VM 200  
HTTP server  
  See IBM HTTP Server

## I

IBM HTTP Server  
  LDAP protection for 121  
    customizing httpd.conf 121  
    customizing slapd.conf 122  
  WebSphere troubleshooter 121  
IFAPRDxx parmlib member  
  relation to dynamic enablement 12  
ISPF, WebSphere for z/OS Customization dialog 190

## K

Kerberos  
  See Security Server Network Authentication Service  
keyboard 217

## L

LAN Server  
  NFS  
    description 64  
LANs  
  LAN A description 60  
  LAN B description 61  
  LAN C description 62  
  token-ring backbone description 60  
LDAP Server 125  
  configuration overview 125  
  enabling for Kerberos authentication 134  
  our experiences 134  
  native authentication  
    enabling IBM HTTP Server to use 130  
    overview of HTTP Server protection using 130  
  native authentication, enabling 126  
  modifying LDAP entries 128  
  updating the schema 126  
  updating the slapd.conf configuration file 127  
  referral between LDAP databases 132  
  defining a referral in z/OS LDAP Server 133  
  setting up an LDAP server on Windows NT 132  
  testing the referral function 133  
  resolving malformed DNS 137  
Linux  
  running on IBM mainframe servers 193  
  setting up guests under z/VM 193  
    building the first Linux instance 199  
    creating a VM user ID for the first Linux  
    guest 194  
    creating and tailoring additional Linux  
    instances 200  
  preparing the DASD 194

Linux (*continued*)  
  setting up guests under z/VM (*continued*)  
    setting up HiperSockets communications 200  
    setting up IP network connectivity 195  
  setting up HiperSockets communications under  
  z/VM 200  
    adding HiperSockets links to the Linux virtual  
    machine 202  
    configuring the network interface under  
    Linux 203  
    defining the iQDIO hardware configuration 201  
    experiences with HiperSockets  
    communications 205  
  setting up LPAR CPU management for the z/VM  
  partition 205  
LPAR CPU management  
  setting up for non-z/OS partitions 205  
  setting up for the z/VM partition 205

## M

MQSeries  
  clusters 41  
  SDSF 41  
  setting up 31  
  shared queues 31  
    configuration 32  
    coupling facility structures for 37  
    DB2 environment 33  
    initialization input data sets 38  
    MQSeries entries to DB2 39  
    MQSeries-CICS bridge queues 40  
    system parameter module 40

## N

naming conventions  
  CICS and IMS subsystem jobnames 14  
Network Authentication Service  
  See Security Server Network Authentication Service  
networking  
  configuration 57  
  ATM LAN emulation 59  
  Ethernet LAN 58  
  token-ring LAN 59  
  workloads 65  
NFS  
  comparing z/OS NFS and LAN Server NFS 64  
  migrating to the OS/390 NFS 64  
Notices 219

## O

OS/390 UNIX System Services  
  managing a zFS 115  
OSA-2  
  ATM feature 57  
  ENTR feature 57, 58, 59  
  FENET feature 57, 58  
OSA-Express  
  ATM feature 57

OSA-Express (*continued*)  
FENET feature 57, 58  
Gigabit Ethernet 57  
Gigabit Ethernet feature 58

## P

Parallel Sysplex  
hardware configuration 3  
Parallel Sysplex automation 51  
parmlib members  
related to z/OS V1R3 and z/OS.e V1R3 207  
performance  
*See also* RMF  
RMF reports 209  
z/OS 29  
PKI Services 157  
configuring the system for 160  
configuring the UNIX runtime environment 162  
running IKYCVSAM to create VSAM data sets 166  
running IKYSETUP 160  
starting the HTTP server instances 165  
starting the PKI Services daemon 167  
tailoring the LDAP configuration 163  
tailoring the PKI Services configuration for LDAP 166  
updating the HTTP server configuration 164  
customizing 169  
end-user Web pages 169  
installing and configuring prerequisite software 157  
IBM HTTP Server 157  
LDAP Server 159  
OCEP 159  
OCSF 158  
using 170  
creating a five-year PKI SSL server certificate 174  
creating a one-year SAF browser certificate 172  
creating a one-year SAF server certificate 170  
creating a two-year PKI browser certificate for authenticating to z/OS 173  
end-user Web pages 170  
installing the CA certificate into the browser 170

## R

RMF  
Monitor I Post Processor Summary Report 209  
Monitor III Online Sysplex Summary Report 209  
Workload Activity Report in WLM Goal Mode 210

## S

SA OS/390  
*See* Parallel Sysplex automation  
Security Server LDAP Server  
*See* LDAP Server  
Security Server Network Authentication Service 139  
resolving setup problems 139  
setup experiences 139

Security Server Network Authentication Service (*continued*)  
verifying our customization 141  
shared queues 31  
configuration 32  
coupling facility structures for 37  
DB2 environment 33  
initialization input data sets 38  
MQSeries entries to DB2 39  
MQSeries-CICS bridge queues 40  
system parameter module 40  
shell variables, using to control file tagging on redirection 153  
shortcut keys 217  
software  
configuration overview 12  
sysplex configuration 12

## T

tasks  
migrating to z/OS  
overview 21  
migrating to z/OS V1R3 21  
high-level migration process 21  
other migration activities 22  
migrating to z/OS.e V1R3 23  
high-level migration process 24  
other migration activities 26  
token-ring LAN  
backbone description 60  
configuration 59  
LAN A description 60  
LAN B description 61  
LAN C description 62  
troubleshooter  
WebSphere Application Server for z/OS and OS/390 177

## U

UNIX  
*See* z/OS UNIX System Services  
URLs  
referenced by our team 215

## V

VTAM  
configuration 14

## W

Web sites  
used by our team 215  
WebSphere Application Server 177  
WebSphere Application Server for z/OS and OS/390  
enabling on a sysplex 181  
customizing base z/OS functions 182

- WebSphere Application Server for z/OS and OS/390
    - (*continued*)
      - enabling on a sysplex (*continued*)
        - defining the second WebSphere for z/OS system 184
        - planning for sysplex 182
        - preparing the security system 182
        - running the installation verification program 186
        - setting up data sharing 182
        - setting up LDAP for other systems in the sysplex 183
        - updating the TCP/IP configuration 183
      - migrating to WebSphere for z/OS V4.0.1 188
        - migration steps 189
        - other migration experiences 192
        - running the customization ISPF dialog 190
      - migrating Web applications to the V4.0 plugin 180
      - migrating Web applications to the WebSphere for z/OS V4.0 plugin 180
      - packaging and deploying Web applications using WAR files 177
        - experiences with 178
        - sample descriptor file 179
        - steps for 177
      - troubleshooter 177
      - updating configuration for servlets that access MOFW or EJB applications from the V4.0 plugin 187
        - accessing our servlet 188
        - updating the HTTP server startup procedure 187
        - updating the httpd.conf file 188
        - updating the httpd.envvars file 188
        - updating the was.conf file 188
      - using 177
      - where to find more information 192
    - WebSphere troubleshooter 121
    - workload
      - application enablement 17
      - automatic tape switching 16
      - base system functions 16
      - database product 19
      - DB2 batch 20
      - DB2 data sharing 19
      - IMS data sharing 19
      - networking 18
      - networking and application enablement 65
      - sysplex batch 20
      - sysplex OLTP 19
      - VSAM RLS data sharing 19
    - workloads 15
- Z**
- z/OS
  - performance 29
  - summary of new and changed parmlib members for z/OS V1R3 and z/OS.e V1R3 207
  - z/OS Security Server LDAP Server
    - See LDAP Server
  - z/OS UNIX System Services 67
    - access control lists (ACLs) 74
  - z/OS UNIX System Services 67 (*continued*)
    - about ACL terminology 75
    - access ACLs 75
    - access ACLs, working with 76
    - auditing changes to 83
    - base ACL entries 75
    - default ACLs, working with 79
    - directory default ACLs 75
    - extended ACL entries 75
    - file default ACLs 75
    - overview of 74
    - using copytree with 102
    - using cp and mv commands with 102
    - using df and getconf commands to determine file system support for 104
    - using find command with 94
    - using getfacl and ls commands with 86
    - using ISHELL with 112
    - using pax and tar commands with 99
    - using setfacl command with 90
    - using UNIXPRIV class profiles with 106
    - enhancements in z/OS V1R3 67
      - automount enhancement 67
      - BPXPRMxx UNMOUNT option for file system mounts 68
      - colony address spaces outside of JES 69
      - enhanced confighfs command 67
      - mount table limit monitor 71
      - OMVS restart 68
    - managing a hierarchical file system (HFS) 115
  - z/OS V1R3
    - high-level migration process 21
      - applying coexistence service 22
      - IPLing additional z/OS V1R3 images 22
      - IPLing the first z/OS V1R3 image 22
      - updating parmlib 22
      - updating RACF templates 22
    - other migration activities 22
      - additional APARs for z/OS V1R3 23
      - data set naming for JES2 and JES3 libraries 23
      - recompiling REXX EXECs for automation 23
      - running with mixed product levels 22
      - using concatenated parmlib 22
  - z/OS.e V1R3
    - high-level migration process 24
      - IPLing the system 26
      - obtaining licenses for z/OS.e 25
      - updating our IEASYMPT member 26
      - updating our LOADxx member 25
      - updating parmlib 25
      - updating the LPAR name 25
    - other migration activities 26
      - LPAR environment 26
      - removing from MNPS 27
      - removing from TSO generic resource groups 27
      - running with mixed product levels 26
      - updating our ARM policy 27
      - using concatenated parmlib 26
      - using current levels of JES2 and LE 27
    - other migration experiences 28
    - overview 23

z/OS.e V1R3 (*continued*)  
    summary of requirements 28  
z/VM  
    setting up Linux guests under 193  
zFS  
    *See also* OS/390 UNIX System Services  
    setting up 115  
    terminology 115



---

# Readers' Comments — We'd Like to Hear from You

z/OS  
Parallel Sysplex Test Report  
V1R3

Publication No. SA22-7663-05

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



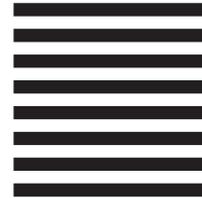
NO POSTAGE  
NECESSARY  
IF MAILED IN THE  
UNITED STATES

# 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 U.S.A.

SA22-7663-05

