z/OS Version 1 Release 2 Implementation

64-bit virtual storage addressing, GRS, SMP/E, WLM, RMF, LE, Kerberos

zSeries File System, JES2, JES3, SDSF, UNIX System Services

RACF, Firewall technology, Cryptographic Services

Paul Rogers
Joaquim e Silva
Roar Framhus
Hirokazu Kakurai
Adriana Leo
Valeria Martins

ibm.com/redbooks

International Technical Support Organization

z/OS Version 1 Release 2 Implementation

May 2002

SG24-6235-00

Take Note! Before using this information and the product it supports, be sure to read the general information in “Notices” on page xi.

First Edition (May 2002)

This edition applies to z/OS Version 1 Release 2 of z/OS (5694-A01), and to subsequent releases and modifications until otherwise indicated in new editions.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

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

Contents

Notices

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

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not 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 U.S.A.

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

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

Any references in this information to non-IBM 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.

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 the International Business Machines Corporation in the United States, other countries, or both:

e (logo)®
Redbooks(logo)™
AFP™
AIX™
AS/400™
AT™
CICS™
CT™
Current™
DB2™
DB2 Universal Database™
DFS™
DFSMSrmm™
DFSORT™
FICON™
Footprint™
Hiperspace™
IBM™
IMS™
Infoprint™
iSeries™
Language Environment™
MQSeries™
Multiprise™
MVS™
MVS/ESA™
NetView™
OS/2™
OS/390™
PAL™
Parallel Sysplex™
PR/SM™
PR/SM™
RACF™
Redbooks™
RMF™
S/390™
S/390 Parallel Enterprise Server™
SecureWay™
SP™
SP1™
SP2™
S/390 Parallel Enterprise Server™
VTAM™
WebSphere™
XT™
z/Architecture™
z/OS™
zSeries™
3090™
3890™
400™

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

Lotus®
Word Pro®
Notes™
NetView™

The following terms are trademarks of other companies:

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

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

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.

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

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

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

Preface

This IBM Redbook highlights many new enhancements made in z/OS Version 1 Release 2. You can use this information to help you install, tailor, and configure z/OS Version 1 Release 2.

It will give you an understanding of a new architecture for 64-bit virtual storage addressing. Other topics discussed are the enhancements to z/OS base control program (BCP) elements; z/OS JES2 Version 1 Release 2, a new zSeries File System, Spool Display and Search Facility (SDSF), UNIX System Services, SMP/E, Workload Manager, RMF, z/OS JES3 Version 1 Release 2, and Language Environment.

This redbook also describes several SecureWay Security Server functional enhancements to RACF, firewall technology, z/OS cryptographic services, and Security Server Kerberos.

It is intended for system programmers and administrators responsible for customizing and installing z/OS.

The team that wrote this redbook

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

Paul Rogers is a Consulting IT Specialist at the International Technical Support Organization Poughkeepsie Center. He writes extensively and teaches IBM classes worldwide on various aspects of OS/390. Before joining the ITSO 12 years ago, he worked in the IBM Installation Support Center (ISC) in Greenford, England providing OS/390 and JES support for IBM EMEA.

Joaquim e Silva is a Senior Systems Engineer at Standard Bank, South Africa. He has 18 years of experience in the OS/390 systems programming field. His areas of expertise include OS/390, Parallel Sysplex, DFSMS, and related products. He has contributed to several redbooks about previous releases of OS/390.

Roar Framhus is a Senior Systems Engineer in Norway. He has more than 30 years of experience in the large computer field, including 19 years at IBM. His areas of expertise include S/390 hardware and software. He has written extensively on S/390 topics.

Hirokazu Kakurai is an IT specialist in IBM Japan Systems Engineering Co. He has 7 years of experience with IBM. His areas of expertise include OS/390, Parallel Sysplex, and WLM. He has written extensively on WLM topics.

Adriana Leo is a Senior Systems Analyst at Banco do Brasil S.A. She has 11 years of experience in the Information Security field. She holds a degree in Electrical Engineering and a Master’s degree in Communication from the Federal University of Pernambuco. Her areas of expertise include communication technologies, data security, internet security, security protocols, PKI, and cryptography. She has written extensively on security topics.

Valeria Martins is a Senior Systems Analyst at Banco do Brasil S.A. She has 7 years of experience in the large computer field. She holds a degree in Computer Science from the State University of Rio de Janeiro and an MBA degree in Networking from the University of São Paulo. She is an OS/390 systems programmer. Her areas of expertise include OS/390, Parallel Sysplex, DFSMS, and related products. She has contributed to several redbooks about the IBM SecureWay Security Server.

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

Franco Meroni
IBM Italy

Richard Conway
International Technical Support Organization, Poughkeepsie Center

Robert Haimowitz
International Technical Support Organization, Poughkeepsie Center

Special notice

This publication is intended to help system programmers and IBM personnel to understand the changes made to the operating environment in z/OS Version 1 Release 2. The information in this publication is not intended as the specification of any programming interfaces that are provided by z/OS Version 1 Release 2. See the PUBLICATIONS section of the IBM Programming Announcement for z/OS Version 1 Release 2 for more information about what publications are considered to be product documentation.

Comments welcome

Your comments are important to us!

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

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

  `ibm.com/redbooks`

- Send your comments in an Internet note to:

  `redbook@us.ibm.com`

- Mail your comments to the address on page ii.

1

z/OS Version 1 Release 2 overview

This chapter contains an overview of the new functions available in z/OS Version 1 Release 2.

It briefly describes the following enhancements to the existing z/OS components:

- z/OS 64-bit virtual support
- z/OS base control program (BCP)
- z/OS JES2 Version 1 Release 2
- Spool Display and Search Facility (SDSF)
- zFS file system
- UNIX System Services
- SMP/E
- Workload manager
- RMF
- z/OS JES3 Version 1 Release 2
- SecureWay security server LDAP
- RACF
- Firewall technology
- z/OS Cryptographic services
- Security server Kerberos

z/OS Version 1 Release 2 runs on the following IBM servers:

- A z900 or comparable server
- S/390 Parallel Enterprise Servers, Generation 5 (G5) and Generation 6 (G6) models
- All models of the Multiprise 3000 Enterprise Server

z/OS Version 1 Release 2, together with the IBM zSeries 900 or comparable server, continues to support the business needs of customers by offering the following enhancements:

- Security enhancements that help bolster protection from outside attack by providing digital certificate and key ring administration, and allow greater support for “Smart Cards” and banking standards
- msys for operations enhancements
- Extended usage of msys for setup for a z/OS configuration
- Quicker and more seamless integration of software delivery and installation
- Increased productivity and fewer required skills in managing resources
- A new high-speed low-latency TCP/IP communication (HiperSockets) between logical partitions that will encourage deployment of new Linux and z/OS applications on the z900 servers

> **Restriction:** APAR OW51521 enforces the selection of the proper architecture level beginning with z/OS 1.2. If you are running on a 9672 G5, G6, or Multiprise 3000: ARCHLVL=1 should be specified or allowed to default. If ARCHLVL=2 is specified, then you will receive the following message:
>
> ```
> IEA368I INVALID RECORD IN LOADxx.  FIRST 17 BYTES ARE: 'ARCHLVL  2       '
> ```
>
> The system ignores the incorrect record and system initialization continues in ESA/390 mode. And if you are running on a zSeries (z900): ARCHLVL=2 should be specified or allowed to default. If ARCHLVL=1 is specified, then you will receive the following message:
>
> ```
> IEA368I INVALID RECORD IN LOADxx.  FIRST 17 BYTES ARE: 'ARCHLVL  1       '
> ```
>
> The system ignores the incorrect record and system initialization continues in z/Architecture mode. IBM recommends that the ARCHLVL statement be removed from any LOADxx member beginning with z/OS 1.1. The proper architecture level default will be selected at IPL time.

1.1 64-bit virtual support

In z/OS Version 1 Release 2, the initial basic 64-bit virtual storage management is supported. Assembler programs can take advantage of this basic 64-bit virtual support to obtain virtual storage above 2 GB for storing and manipulating data. The target program must reside in 31-bit-addressable storage, but application data may reside anywhere in 64-bit-addressable storage.

z/OS components that have been enhanced are:

- The Interactive Debug Facility of the High Level Assembler Toolkit Feature (ASMIDF) is enhanced to support debugging of programs running in 64-bit addressing mode.
- Coupling Facility access services support the use of 64-bit virtual storage for buffers used in transferring data to and from the Coupling Facility.
- EXCP services support the use of 64-bit virtual storage for buffers used in transferring data to and from tape or DASD devices.
- The OSA-Express device driver and the OSA-Express adapters are enabled to support 64-bit real storage.

This initial 64-bit virtual storage support offers an opportunity for software products that can make use of large virtual storage to relieve virtual storage constraints and to enhance data caching capabilities.

High level language support

The C/C++ program execution environment supports the following:

- C/C++ Compiler
- LE C/C++ run time library
- UNIX System Services (syscall layer, file system, shell/utilities, shmat, mmap)
- TCP/IP and file systems support 64-bit data
- dbx debugger
- A selected small number of z/OS system services 64-bit APIs

1.2 Base control program (BCP) enhancements

Changes to the BCP in Release 2 are in the following functional areas:

1.2.1 Global resource serialization (GRS)

GRS is a required component of a Parallel Sysplex environment. GRS ENQ/DEQ requests can be modified by specifying resource names in RNLs. Prior to the GRS wildcard support in z/OS V1R2, GRS resource names in RNLs could match resource requests either exactly or by having the prefix specification in the RNL match the resource request.

The use of system cloning techniques and the fact that more operating system components are using GRS services means that more resources need to be managed with RNLs. This has led to a growing number of resources for which prefix matching is insufficient. As a result, the RNL matching mechanism must be more flexible. The GRS RNL wildcard support adds a new RNL specification that allows the use of wildcard characters ("*" and "?") within resource names to provide an extremely flexible means to modify ENQ/DEQ requests.

Furthermore, the RNL exit (ISGGREX0), an alternate means to modify ENQ requests, has been removed. It has been replaced with a more flexible dynamic exit (ISGNQXIT). The new exit allows the modification of resource QNAME, RNAME, SCOPE, or UCB address (for a RESERVE), or the bypassing of RNL processing entirely. Using the dynamic exits facility allows for this additional flexibility and helps avoid the possibility of a poorly-coded exit causing an outage or resource integrity errors.

1.2.2 Unconditional log close

If the OPERLOG and printer consoles are not available, it is impossible to deactivate the SYSLOG. If the SYSPLG fails and is suspended, MVS saves messages until the hardcopy becomes available again. This may result in an outage because all the available WTO storage becomes filled with messages awaiting hard copy. This support in Release 2 provides an unconditional option on the **`vary hardcpy,off`** command, which allows a WRITELOG CLOSE to be accepted and the SYSLOG to be deactivated.

1.3 JES2 Version 1 Release 2

Greater than 64K job support is now supported, which allows an installation to have their job number range grow from 65,534 jobs to 999,999 jobs.

Long running job support gives an installation the ability to obtain (spin off) their JESlog data sets prior to job completion. Currently, an installation has to cancel a job that runs for a long period of time to obtain the job's JESlog data sets. This support allows an installation to keep these jobs running indefinitely.

1.4 Spool Display and Search Facility (SDSF)

The following new enhancements are available with SDSF with z/OS Version1 Release 2:

- Improvements to the Log Title Line
- Removal of the *PROCESS statements
- Single quote on print panel
- UNIT support on print panel
- Restructure of the UCLIN Jobs
- Add report class to DA panel
- Make enclaves visible to operators
- SDSF displays multiple system affinities
- SDSF together with RMF captures enclave SRBs for DB2
- Maintaining filter lists between SDSF sessions
- Improving the performance of SDSF with OPERLOG is addressed
- A FIN APAR, PQ41542, fixes a problem with "SJ" when it is issued against a job that has USER=&xxxxx on the job card

Workload Manager (WLM) and SDSF are enhanced to allow operators or system programmers to change the service characteristics of work units running as WLM enclaves. This provides improved operational management of distributed DB2, scalable Web Server, and WebSphere 4.0 work requests.

z/OS SDSF requires the following operating system environment:

- BCP at z/OS Version 1 Release 2 in z/Architecture mode
- JES2 at OS/390 Release 7 or higher

1.5 zFS file system

Workload usage patterns of data vary; therefore, the requirements for the underlying data store can vary greatly as well. To meet the changing needs of new workloads, an additional file system to be used with z/OS UNIX System Services is provided. The zSeries zFS file system is a UNIX file system like the Hierarchical File System (HFS), but zFS is not intended to replace HFS. HFS is still assumed to be used for the root file system.

This enhanced file system is complementary to the existing HFS. In z/OS Version 1 Release 2, it depends on the Hierarchical File System to provide the root file structure from which the zFS file systems can be mounted. It also has different system management considerations for installation, backup, and recovery. Migration of data from the HFS to the zFS can be as simple as drag and drop or a batch copy command.

Accessing zFS data from local UNIX applications, combined with the ability to access zFS data from Windows systems (using the SMB server) and UNIX systems (using the NFS server), enables z/OS to provide a more robust file sharing environment.

1.6 UNIX System Services support

To better enable the porting of applications to z/OS UNIX Systems Services platforms, z/OS Version 1 Release 2 supports the following:

- Enhanced ASCII support

  Enhanced ASCII support simplifies the porting of applications from ASCII platforms to a UNIX System Services environment in z/OS.

- ANSI '98 C++ Standard compliance

  ISO C++ 1998 Standard language level, including the Standard Template Library (STL), enables you to more easily port C++ applications from ASCII platforms to a UNIX environment in z/OS.

These enhancements make it easier to port ASCII applications from other UNIX platforms to z/OS UNIX System Services.

System operators are able to manage the z/OS UNIX System Services environment and UNIX file systems more effectively by using several new and updated commands.

1.7 SMP/E enhancements

SMP/E contains the following functional enhancements:

- The ability to install software products and service directly from a network source, such as the Internet. This reduces the tasks and time required to install software delivered electronically.
- The ability to use SMPPTS “spill” data sets to contain the overflow when the SMPPTS data set is full allows multiple physical data sets to be used for a single logical data store.
- SMP/E consolidates and summarizes the HOLD information encountered during APPLY and ACCEPT command processing, reducing the time needed to research HOLD information when installing software service.
- By eliminating the tasks involved when recovering from an overflowing SMPPTS data set, the new SMPPTS spill data set support allows users to install software service with less intervention and less management of the SMP/E environment.
- To assist users in researching and acting upon important HOLD information, SMP/E will consolidate and summarize the HOLD information encountered during APPLY and ACCEPT command processing. The new Holddata report reduces the time needed to research HOLD information when installing software service. The report contains the actual text of the HOLD, thereby saving a user from having to find this information elsewhere.

1.8 Workload Manager support

You are also encouraged to implement Workload Manager (WLM) goal mode with the release of z/OS. Goal mode is required to get the most benefit from the Intelligent Resource Director (IRD) functions. WLM in goal mode continues to grow in its role and importance.

Each new release of the operating system and supporting subsystems brings further exploitation of WLM goal mode for improvements and efficiencies in system performance and workload balancing.

Goal mode is critical to the implementation of many of the strategic solutions to be delivered over the next one to two years.

Attention: z/OS Version 1 Release 2 will be the last release to support WLM compatibility mode. Goal mode will be the only supported mode starting with z/OS Version 1 Release 3, scheduled for the first half of 2002.

A Goal Mode Migration Aid tool to help get to goal mode can be downloaded from:

http://www.ibm.com/servers/eserver/zseries/zos/rmf/rmfhtmls/rmftools.htm#spr_goal

WLM enhancements

The unique workload and self-optimizing capabilities provided by WLM and the Intelligent Resource Director (IRD) allow you to define response goals based on business priorities. z/OS can handle unpredictable workloads, and allows high CPU and I/O utilization, while still meeting response goals with minimal human intervention for setup and operation.

The new workload and LPAR management enhancements create an effective automatic load-balancing capability.

zSeries workload management is now enhanced to also allow non-z/OS partitions, in particular Linux images. The z/OS WLM then manages the CPU resources given to these partitions based on their relative importance compared to the other workloads running in the same LPAR cluster.

WLM provides a new mechanism to dynamically determine the number of server tasks that should be run in a server address space. This removes the user responsibility of monitoring and tuning to react to changing workloads.

Enclave improvements allow subsystems that execute work on behalf of the same single business unit of work (enclave) to protect the transaction from accidental deletion by another subsystem.

In-real swap provides a more efficient algorithm to recover fixed real storage from a “swappable” address space. Real storage can now be recovered without swapping allowing for faster storage reconfigurations.

WLM provides a new service that allows a work manager to preserve a temporary affinity with a specific server region across multiple independent work requests. WebSphere V4.0 uses this function to create objects which live across multiple transactions on behalf of the same client.

1.9 RMF enhancements

RMF has been enhanced to support many of the new features available in z/OS Version 1 Release 2 as follows:

- An RMF Partition Data report helps you understand how much of the defined capacity an LPAR is consuming.
- Management of zSeries cryptographic hardware is made easier by the ability to identify cryptographic bottlenecks based on managed workloads. RMF helps customers understand Cryptographic Coprocessor usage and identifies the workloads that use or experience delays when trying to use the Cryptography Coprocessor, which supports capacity planning.

1.10 JES3 Version 1 Release 2

Greater than 64K job support allows an installation to have their job number range grow from65,534 jobs to 999,999 jobs.

Long running job support gives an installation the ability to obtain (spin off) their JESlog (JESYSMSG and JESMSGLG) data sets prior to job completion. Currently, an installation has to cancel a job that runs for a long period of time to obtain the job's JESlog data sets. This support allows an installation to keep these jobs running indefinitely. Simplified processing requirements for the JES3 global address space is made available by moving message processing to the user's address space.

Support is provided in this release of JES3 to allow spool data sets, JCT spool data sets, or checkpoint data sets to reside on a volume containing more than 32,767 cylinders.

1.11 Kerberos support

z/OS Version 1 Release 2 provides new enhancements to the Kerberos infrastructure that allows users to be authenticated across multiple systems. Additional mechanisms are provided to help protect systems from attacks.

z/OS provides a Kerberos credential server and Kerberos application services. These functions are enhanced with the following:

- Ability to administer the Kerberos registry information
- Stronger encryption
- Automated restart across TCP/IP network outages
- Improved performance in a Parallel Sysplex environment

Several z/OS e-business services are enhanced in z/OS Version 1 Release 2 with support for Kerberos third party authentication as follows:

- Lightweight Directory Access Protocol (LDAP) directory client server
- The z/OS UNIX System Services (USS) versions of FTP, Telnet, and RSH

1.12 LDAP support

LDAP directory service enhancements are provided to increase usability, performance, and integration into security-aware e-business environments. An LDAP Configuration Utility automates basic setup and can be used by any customer. The LDAP Server allows for more clients to be concurrently connected.

The LDAP SDBM function enhances the capability to manage RACF-defined users and groups using the LDAP protocol. These improvements simplify LDAP Client setup by providing the following:

- An ability to bind to LDAP Servers (on and off z/OS) using Kerberos credentials for improved security
- Client-side caching of search results for improved performance of some searches
- The ability to find an LDAP server via information in a Domain Name System (DNS) server without knowing the LDAP server's host name or IP address in advance

1.13 Intrusion Detection Services (IDS)

Firewalls can provide a level of protection against outside attacks. They cannot provide protection when the attack is from within or when end-to-end encryption is employed. The host-based Intrusion Detection Services (IDS) provided in z/OS Version 1 Release 2 complement network-based IDS sensors and scanners. It can discard attacking packets before they cause damage, discard packets exceeding established thresholds, and limit the number of connections from overzealous users. IDS also provides the following:

- Event recording and reporting, including standalone reporting of IDS events (attacks) to a console and the SYSLOG.
- A new specialized IDS packet trace for off-line analysis, and statistics gathering baseline and exception reporting.

1.14 Cryptographic support

New banking standards and unique customer applications require continuing additions of new cryptographic functions in both hardware and software. z/OS Version 1 Release 2 adds support for VISA, Europay and the functions needed for ZKA certification. Release 2 also adds cryptographic functions needed by applications that personalize smartcards for use in Point-of-Sale (POS), Debit, and Stored Value applications.

For unique customer applications, the PCI cryptographic co-processor will support the loading of customized cryptographic functions on zSeries 900 and S/390 G5/G6 processors. With Release 2, zSeries PCI cryptographic co-processors, and under a special contract with IBM, customers will gain the flexibility to define and build customized cryptographic functions themselves.

2

z/OS 64-bit virtual support

This chapter describes the support for z/OS V1R2 64-bit virtual storage management, specifically:

- 64-bit addressing
- Implementing 64-bit virtual support
- Using virtual storage above 2 GB
- Controlling virtual syorage usage
- I/O operations above 2 GB
- Dumping virtual storage above 2 GB
- z/Architecture considerations
- 64-bit virtual storage restrictions

2.1 64-bit virtual addressing

The ESA/390 architecture, with its 31-bit real and virtual addressing scheme, limited the amount of real storage that could be assigned to an OS/390 image and the size of an OS/390 address space to 2 Gigabytes (GB), or 2**31 addresses.

In order to support the growth in e-business with large numbers of users and transactions, a 64-bit virtual addressing scheme will be introduced over time. The first basic support in the z/OS operating system for 64-bit virtual addressing is introduced with z/OS V1R2.

OS/390 V2R10 and the z/Architecture of the IBM 2064 processors introduced 64-bit real storage addressing. The 64-bit real storage support provides for up to 256 GB of central storage to be configured to a z/OS image. OS/390 V2R10 and z/OS V1R1 supports up to 128 GB of real storage when running in z/Architecture mode on an IBM 2064.

z/OS Version 1 Release 2 introduces the initial support for 64-bit virtual addressing. Programs can access the virtual storage area above the 2 GB address. This basic 64-bit virtual storage support is the foundation for the 64-bit z/OS operating system virtual storage infrastructure. At a later stage this basic support will be exploited by high-level languages and middleware.

z/OS and its predecessor operating system have always provided a capability to scale horizontally. Users have the ability to scale by adding additional CICS regions or building the application to use large numbers of data spaces and address spaces.

Most new e-business applications are being built in C/C++ or JAVA, which requires the ability to scale vertically—meaning larger address spaces—instead of horizontally. The user private area can now extend up to 2**64, as shown in Figure 2-1.

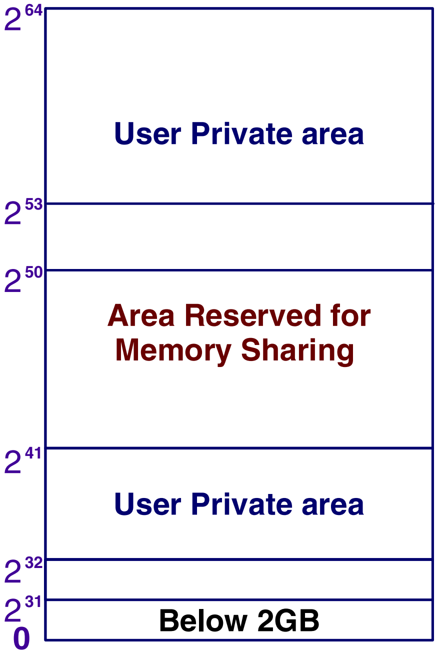


Figure 2-1 Address space virtual storage areas

2.1.1 Size and number notation

With the introduction of 64-bit virtual addresses we are going into a new order of magnitude of numbers. We feel it is appropriate to refresh your mind on the names and raw sizes of numbers we now will start using.

The letters K, M, G, T, P, and E denote the multipliers 2**10, 2**20, 2**30, 2**40, 2**50, and 2**60, respectively. The letters are borrowed from the decimal system and stand for Kilo (10**3), Mega (10**6), Giga (10**9), Tera (10**12), Peta (10**15), and Exa (10**18). Note that Exa is 1 followed by 18 zeroes.

In the z/Architecture world these multiplier letters do not have the decimal meaning but instead represent the power of 2 closest to the corresponding power of 10. Table 2-1 shows the names and the decimal values of these multipliers.

Table 2-1 Multiplier decimal values

Symbol	Power of 2	Decimal Value
Kilo (K)	2**10	1024
Mega (M)	2**20	1,048,576
Giga (G)	2**30	1,073,741,824
Tera (T)	2**40	1,099,511,627,776
Peta (P)	2**50	1,125,899,906,842,624
Exa (E)	2**60	1,152,921,504,606,846,976

Storage examples

The following are some examples of the use of K, M, G, T, and E:

- 2,048 can be expressed as 2K
- 4,096 can be expressed as 4K
- 65,536 can be expressed as 64K
- 2**24 can be expressed as 16M
- 2**31 can be expressed as 2G
- 2**43 can be expressed as 8T
- 2**64 can be expressed as 16E

We use a number and a multiplier letter combination xB, for example 1 KB, to tell how many bytes we are talking about; here we mean 1,024 bytes.

Power of 2 values

With S/390 and z/Architecture, we are used to the power of 2 notation. In Table 2-2 we show you some often-used numbers with the power of 2 notation and its corresponding decimal and hexadecimal value.

Table 2-2 Power of 2 for some often-used numbers

Power of 2	Decimal Value	Hexadecimal value
2**7	128	00000080
2**8	256	00000100
2**12	4096	00001000
2**16	65,536	00010000

Power of 2	Decimal Value	Hexadecimal value
2**24	16,777,216	01000000
2**31	2,147,483,648	80000000
2**32	4,294,967,296	00000001_00000000
2**33	8,589,934,592	00000002_00000000
2**48	281,474,976,710,656	00001000_00000000
2**64	18,446,744,073,709,551,616	1_00000000_00000000

Remember that addresses start at 0.

2.1.2 The 64-bit virtual address space

Over the years the size of the MVS address space has grown with the introduction of new architectures. There were two distinct address space sizes before the 64-bit virtual address space. The address space of the 1970s provided 16 megabyte of virtual addressing with 24-bit addressing. In 1981, Extended Architecture (XA) was introduced with 2 gigabyte address spaces provided by 31-bit addressing. This was 128 times the size of the original 16 MB address space.

To get an overview of the size of the numbers associated with this 64-bit support, see Table 2-2 for power of two values for some numbers often used with computers. As you can see from this table, the 370/XA 2 GB 31-bit address space was 128 times larger than the S/370 16 MB 24-bit address space.

Now, with z/OS V1R2 and the z/Architecture of the IBM zSeries processors, you get a virtual address space size of 2 **64, or 16 exabytes (16 EB). This is eight thousand million (a thousand million is called a billion in the USA, a milliard in other places) times the size of the 2 GB XA address space.

All programs written for 24- or 31-bit addressing mode will continue to run without change. The area below the bar is mapped as before, and is totally compatible.

Virtual storage layout

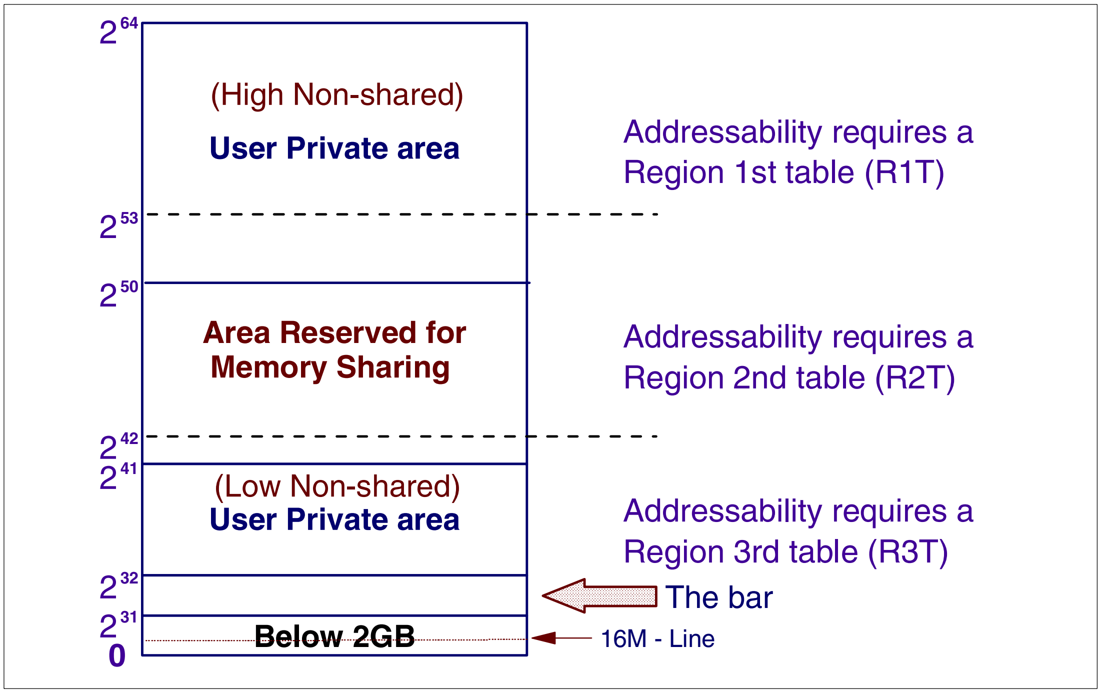
The layout of the storage areas for an address space has the system areas existing both below and above 16 megabytes, providing an environment that can support both 24-bit and 31-bit addressing. However, each area and its counterpart above 16 megabytes can be thought of as a single logical area in virtual storage. This concept is maintained in order to support programs that execute in 24-bit, 31-bit, and 64-bit architecture modes.

Each virtual address space still consists of:

- The common area below 16 megabytes
- The private area below 16 megabytes
- The extended common area above 16 megabytes
- The extended private area above 16 megabytes

The area that separates the virtual storage area below the 2 GB address from the user private area is called the bar, as shown in Figure 2-2 on page 13. This area is also 2 GB.

In a 64-bit virtual storage environment, the terms “above the bar” and “below the bar” are used to identify the areas between 2**31 and 2**64-1, and 0 and 2**31-1, respectively. For example, any address in the range 0 to 7FFFFFFF would be below the bar, and addresses in the range FFFFFFFF to 7FFFFFFF_FFFFFFFF would be above the bar. This is basically an alteration to the 2 GB 31-bit terminology that related “below the line” to 24-bit storage, and “above the line” to 31-bit addresses.



2^{64}
(High Non-shared)
User Private area
Addressability requires a Region 1st table (R1T)
2^{53}
2^{50}
Area Reserved for Memory Sharing
Addressability requires a Region 2nd table (R2T)
2^{42}
2^{41}
(Low Non-shared)
User Private area
Addressability requires a Region 3rd table (R3T)
2^{32}
The bar
2^{31}
Below 2GB
16M - Line
0

Figure 2-2 Address space memory map

Address memory map

The address memory map, shown in Figure 2-2, is described as follows:

0 - 16M 16M is still referred to as the line. Below the line can be addressed with a 24 bit address.

0 - 231** Above the line requires a 31 bit address and does not usually refer to storage beyond 2G.

231 - 2**32** From 2G to 4G is considered the bar. Below the bar can be addressed with a 31 bit address. Above the bar requires a 64 bit address.

The bar

Just as the system does not back the page at 7FFFF000 in order to protect programs from addresses which can wrap to 0, the system does not back the virtual area between 2G and 4G. That means that a 31 bit address with the high bit on, will always program check if used in amode 64. Above the bar is divided into 3 areas as follows:

231 - 2**41** The Low Non-shared area starts at 4G and goes to 2T (2**41).

241 - 2**50** The Shared Area starts at 2T (2**41) and goes to 1P (2**50) or higher if requested.

> **Note:** The Shared Area will not be supported until the UNIX and C 64-bit support is added in a future release. It has been left in the picture at this time to show where the support is going.

250 - 2**64** The High Non-shared area starts at 1P (2**50) or wherever the Shared Area ends and goes to 2**64.

User private area

The area above the bar is intended for application data; no programs run above the bar. No system information or system control blocks exist above the bar, either.

There is currently no area above the bar that is common to all address spaces. However, IBM reserves an area above the bar to be used for future enhancements. You can also reserve some area in the virtual storage you allocate above the bar. This area is called the guard area. This area can be used for enhancements of your application in the future.

The *user private area*, as shown in Figure 2-2 on page 13, includes low private (below the line), extended private (above the line), Low Non-shared, and High Nonshared.

As users allocate private storage above the bar, it will first be allocated from the Low Non-shared area. Similarly, as Shared Area is allocated, it will be allocated from the bottom up.

This is done to allow applications to have both private and shared memory above the bar and only need a region 3rd table (R3T). This is important because, as additional levels of region tables are required, it introduces additional machine cycles to perform dynamic address translation (DAT).

If users allocate large private or shared areas above the bar, then Real Storage Manager (RSM) needs to construct region second tables (R2T) or region first tables (R1T) to manage the virtual storage.

2.1.3 Dynamic address translation

Dynamic address translation is the process that maps virtual 4 KB pages to real memory 4 KB frames. It is performed by the Virtual Storage Manager (VSM) component, and is implemented through a table lookup process on a set of translation tables. These tables are called segment tables and page tables. With 31-bit virtual addressing, which gives you a 2 GB address space, virtual storage is divided into 2048 one megabyte segments per address space, and each address space has its own segment table. Each segment in the segment table has its page table, pointing to a maximum of 256 four KB pages, which is 1024 KB or 1 MB, or one segment.

In a 2 GB address space with 31-bit addressing, you only need one segment table and 2048 page tables.

Region tables

In a 16 EB address space with 64-bit virtual storage addressing, there are three additional levels of translation tables, called region tables. They are called region third table (R3T), region second table (R2T), and region first table (R1T). The region tables are 16 KB in length, and there are 2048 entries per table. When the first storage is created above the bar, RSM creates the R3T. The R3T table has 2048 segment table pointers, and provides addressability to 4 TB. When virtual storage greater than 4 TB is allocated, an R2T is created. An R2T has 2048 R3T table pointers and provides addressability to 8 PB. An R1T is created when virtual storage greater then 8 PB is allocated. The R1T has 2048 R2T table pointers and provides addressability to 16 EB. Figure 2-3 on page 15 shows the page table hierarchy and the size of virtual storage each table covers.

RSM only creates the additional levels of region tables when necessary to back storage which is mapped. They are not built until a translation exception occurs. So, for example, if an application requests 60 PB of virtual storage, the necessary R2T, R3T, segment table, and page tables are only created if they are needed to back a referenced page. Up to five lookup tables may be needed by DAT to do translation, but the translation only starts from the table that provides translation for the highest usable virtual address in the address space.

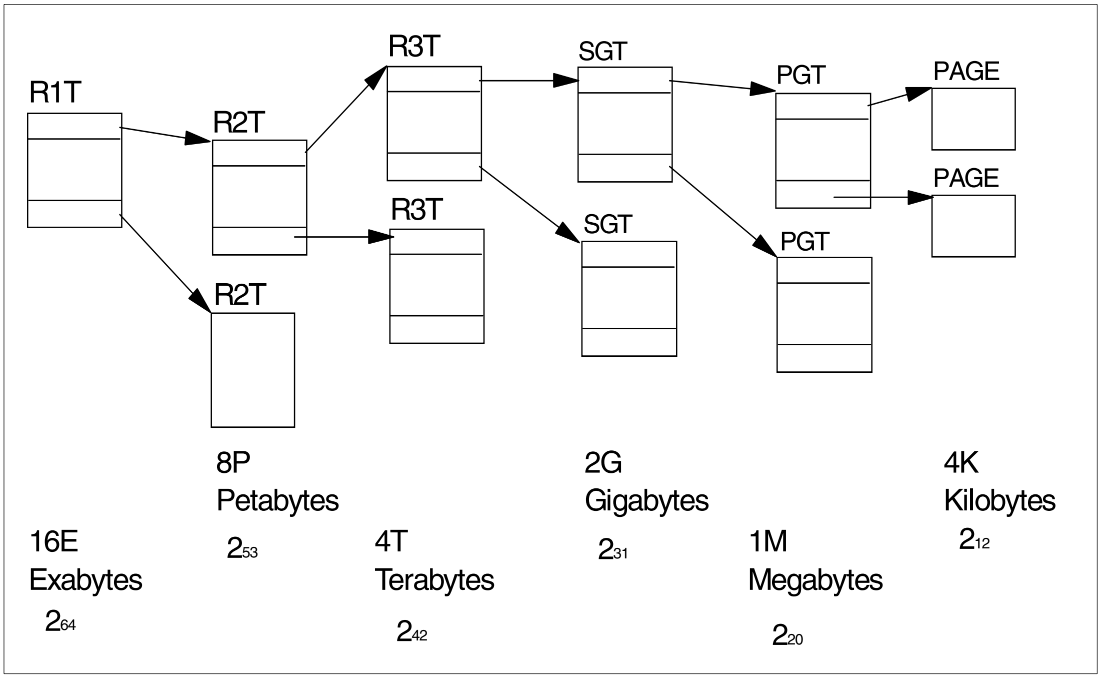


Figure 2-3 DAT with region tables

Segment tables and page table formats remain the same as for virtual addresses below the bar.

2.1.4 The virtual address

The virtual address used in 31-bit addressing consists of three indexes as follows:

- The 11-bit segment index into the segment table
- The 8-bit page index into the page table
- The 12-bit byte index into the page itself

With 64-bit virtual addressing, we now get three more 11-bit region indexes into the three region tables, as shown in Figure 2-4 on page 16.

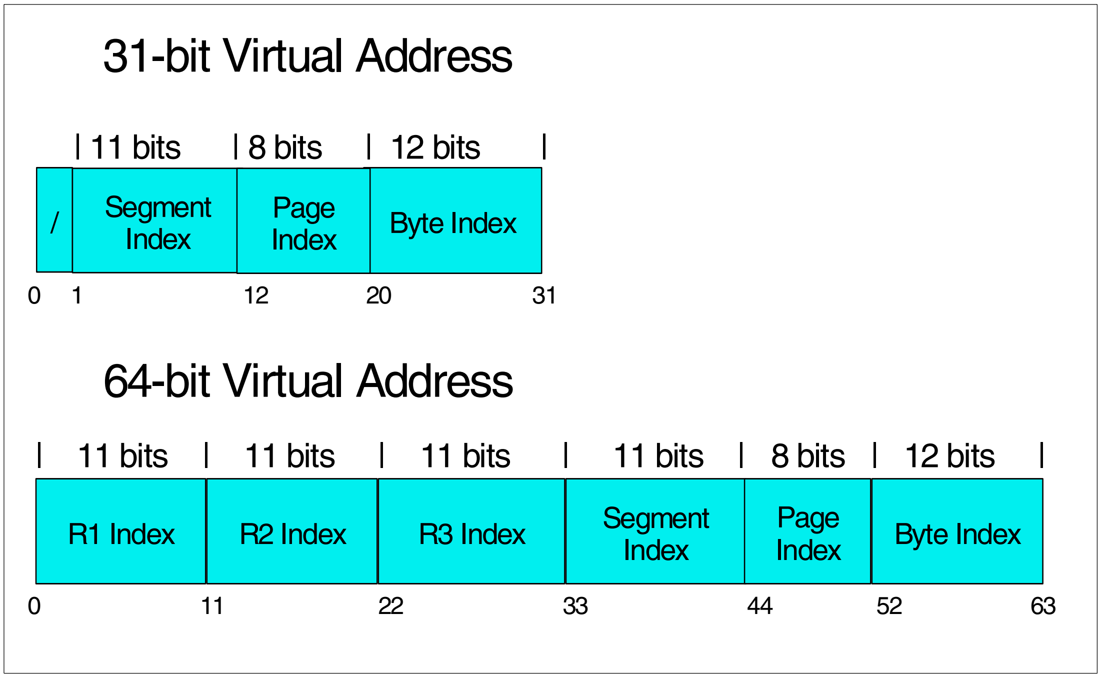


Figure 2-4 Virtual address formats

The 12-bit byte index X'FFF' = 4095 accommodates a 4096 byte page, 0 - 4095. The 8-bit page index X'FF' = 255 accommodates a page table with 256 entries, 0 - 255. The 11-bit segment and region index X'7FF' = 2047 accommodates a region or segment table with 2048 entries, 0 - 2047.

2.1.5 Memory objects

The design point for using virtual storage above the bar is that this storage is managed as objects, and that the external interface is simple. It is distinct from storage below the bar. The exploiters may be language runtime routines with existing low-level storage managers, server-type programs with large storage requirements, and database systems. Unfortunately, size does matter when using large amounts of virtual storage; it must be backed up by real devices.

The z/OS virtual memory above the bar, the virtual line at the 2 GB virtual address, is organized in *memory objects*. A memory object is an area of virtual storage obtained by a program using the IARV64 macro; it resides above the 2 GB address line, the bar, in a 64-bit address space. The system allocates a memory object as a contiguous range of virtual storage addresses in a number of segments.

IARV64 macro

A new system service, IARV64, is introduced to allocate and manage storage above the bar. IARV64 is the only external programming interface provided to allocate and manage storage above the bar. Only data can be stored in virtual storage above the bar. You use the z/Architecture instructions that handle 64-bit registers to access data above the bar. Programs are still loaded below the bar, but they must be running in 64-bit Addressing Mode (AMODE 64) to address the data in virtual storage above the bar. Virtual storage above the bar cannot be used for loading and executing programs; they must be loaded and executed below the bar. z/OS V1R2 does not keep any control information or control blocks above the bar. This area is just for data.

Memory object properties

The system allocates a memory object as a number of 1 MB segments in a contiguous range of virtual addresses. Each segment is one MB in size, and is allocated on a megabyte boundary. A memory object can be as large as allowed by your installation, or as small as 1 MB. There is no micromanagement of virtual storage within a memory object of 1 MB in size. Using the IARV64 macro, a memory object is allocated by a single request and can only be freed in its entirety. Partial freeing is not allowed.

You can only obtain one memory object by a single invocation of IARV64, but you can relate a set of two or more memory objects to each other by specifying a user token. Your program can then later delete all memory objects with the same user token value.

All memory objects are owned by a task, represented by a *task control block* (TCB). A program creates the memory object, but the program does not own the memory object. It is owned by a TCB. If the unit of work that represents the program that created the memory object is a TCB, that TCB is the owner of the memory object. If the unit of work is an SRB, that SRB must assign ownership to a TCB.

A program can assign ownership of the memory object to another TCB. An unauthorized program can assign ownership to a TCB in its TCB hierarchy, up to the jobstep TCB. An authorized creator of memory objects can assign ownership to any TCB within the address space.

2.1.6 Guard area

When a program creates a memory object, it can specify that the memory object is to consist of two different areas: the usable area and the guard area. The guard area is an optional area within a memory object, sometimes also referred to as a hidden area. It is a number of reserved pages, in multiples of megabytes, starting on a megabyte boundary. The guard area is allocated either from the low virtual address of the memory object and upwards, or from the high virtual address and downwards.

The guard area is introduced so that you can reserve an area for future utilization of space for your application. Another way you can use the guard area is to protect yourself from accidentally referencing virtual storage beyond the end of a memory object, and overlaying data in an adjacent memory object. In this case, you should allocate a guard area at high virtual address in your memory object. Similarly, if another program wants to protect its memory object from another program using a memory object at a lower virtual address, it will create a guard area at the lower end of its memory object.

Guard areas cannot be referenced by your program. If you reference the guard area, you will get a program check. The size of the guard area does not count when the system is performing the MEMLIMIT checking, only the usable area does count. The size of the guard area can be changed—for example, to get more usable virtual storage within your memory object. You can increase or decrease the size by using the CHANGEGUARD service. The amount of change is in multiples of one megabyte segments.

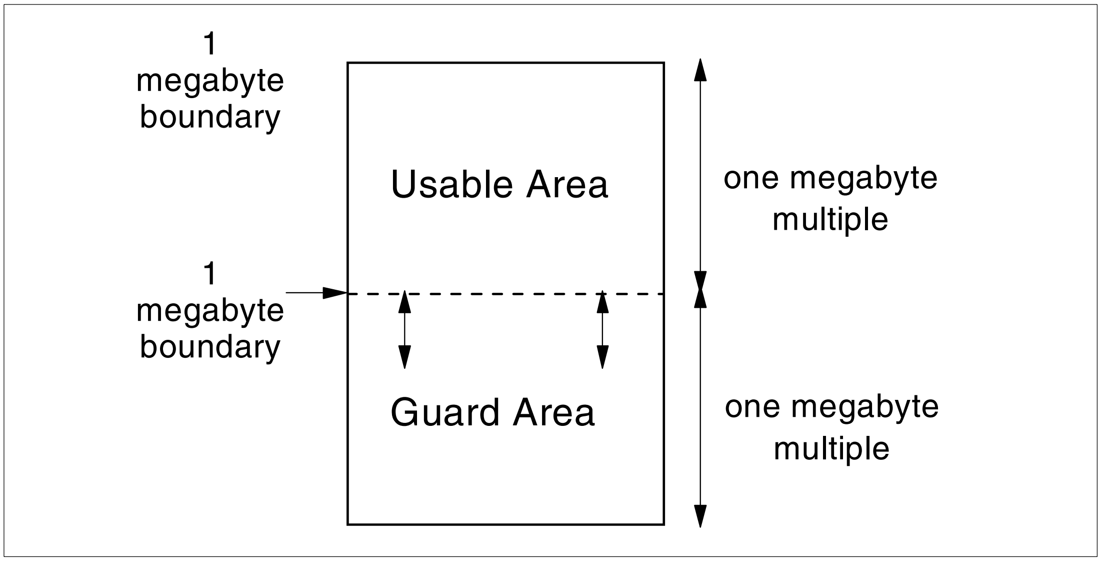


Figure 2-5 A memory object

Memory object operations

There are several services provided by the assembler service macro IARV64. You can think of some of these services like GETMAIN/FREEMAIN or storage services that are used for virtual storage requests below the bar.

A memory object is created by the IARV64 GETSTOR macro. You must specify a total size of the memory object in 1 MB segments. You can also specify the size of the guard area, and whether the guard area is to be at the bottom (low virtual address) or top (high virtual address) of the memory object. By making the request for a memory object conditional, you can avoid an abend if your request cannot be satisfied. The address of the virtual storage associated with the memory object you obtain is returned to your program. If you specified that the guard area should be at the bottom in the memory object, the returned address will point to the guard area; since this is not usable storage for your program, you will get a program check if you reference the guard area.

When a memory object is created, a control block is created which describes the virtual address range and attributes. The translation tables required to map the area are not created immediately. The translation tables are created as required in response to translation exceptions within the usable area of the memory object. These translation exceptions are solved iteratively. Translation exceptions in the guard area result in an abend.

You can help the system manage the real storage frames that back up your pages in a memory object. If you don’t need some pages for a while, you can suggest that the system can make these pages available for stealing and page them out. Later you can suggest that the system start paging them in.

An authorized program can ask the system to fix pages in a memory object, making them unavailable for stealing.

A memory object persists until it is deleted by the DETACH service, or until the task that owns the memory object terminates.

You can relate memory objects to each other by using a user token when you create them. You can then later delete the memory object group by using the user token in the DETACH service.

2.2 64-bit virtual support

This is the first step in introducing 64-bit virtual addressing. This basic support includes:

- 64-bit data addressability within an address space
- Ability to store and manipulate data above 2 GB

The 64-bit virtual storage data addressability within a single address space is available from an assembler program. To use the storage above the bar, a program must change to 64-bit addressing mode (AMODE 64).

There is also an assembler system service, or a macro, IARV64, to allocate and manage virtual storage above the 2 GB virtual address line (the bar) within a single address space. You may think of this as a new GETMAIN service for use above the bar. Any 24-bit or 31-bit assembler program can use this support to get virtual storage above 2 GB, and to store and manipulate data in this area.

IARV64 allows the caller to allocate and manipulate virtual storage above the bar in one megabyte multiples. This allocated virtual storage above the bar is organized in what is called memory objects. The memory objects are allocated and released in sizes of 1 MB chunks.

Programs continue to be loaded and executed below the bar.

2.2.1 Implementing 64-bit virtual

Your system must be loaded in z/Architecture mode to support 64-bit virtual. This is done by placing the ARCHLVL 2 statement in the LOADxx member in your parmlib, SYSn.IPLPARM or SYS1.PARMLIB, as shown in Figure 2-6 on page 20. You should consider placing the LOADxx member in SYS0.IPLPARM, since the system looks for the LOADxx member in this data set first.

The ARCHLVL n statement specifies the mode in which z/OS will run: ARCHLVL 1 means ESA/390 mode, and ARCHLVL 2 means z/Architecture mode. If you do not specify ARCHLVL, the default is ARCHLVL 2, z/Architecture mode, on a zSeries processor, and on other processors the default is ARCHLVL 1, ESA/390 mode.

If you specify ARCHLVL 2 on a processor that does not support z/Architecture mode, a wait state code 088-10 results.

```
NUCLEUS  1
NUCLST   XX
ARCHLVL  2
IEASYM   XX
SYSPLEX  SANDBOX  Y
IODF     ** SYS6     L06RMVS1 01 Y
SYSCAT   SBOX011   MCAT.SANDBOX.VSBOX11
PARMLIB  SYS1.SYSPROG.PARMLIB
PARMLIB  SYS1.PARMLIB
PARMLIB  CPAC.PARMLIB
PARMLIB  SYS1.IBM.PARMLIB
*-----------DEFINITION FOR SC63-------------*
HWNAME   SCZP801
LPARNAME A8
SYSCAT   SBOX111   MCAT.SANDBOX.R10.VSBOX11
*-----------DEFINITION FOR SC64-------------*
HWNAME   SCZP801
LPARNAME A9
SYSCAT   SBOX111   MCAT.SANDBOX.Z02.VSBOX11
*-----------DEFINITION FOR SC65-------------*
HWNAME   SCZP801
LPARNAME A10
SYSCAT   SBOX111   MCAT.SANDBOX.R10.VSBOX11
```

Figure 2-6 LOADxx member for z/Architecture mode

Display architecture mode

You can verify that you system is running in z/Architecture mode by issuing the **Display IPLINFO** command, as shown in Figure 2-7.

```
D IPLINFO
IEE254I  12.29.35 IPLINFO DISPLAY 764
 SYSTEM IPLED AT 19.12.39 ON 05/17/2001
 RELEASE z/OS   01.02.00
 USED LOADS8 IN SYS0.IPLPARM ON 3800
 ARCHLVL = 2   MTLSHARE = N
 IEASYM LIST = XX
 IEASYS LIST = (R3,64) (OP)
 IODF DEVICE 3800
 IPL DEVICE 3833 VOLUME Z02RA1
```

Figure 2-7 D IPLINFO command

Setting addressing mode

When z/OS V1R2 is successfully loaded in z/Architecture mode, a new bit is set in the CVT, CVTV64. When this bit is on, it indicates that the 64-bit support is present.

To reference storage above the bar, the program must have obtained a memory object using the IARV64 system service and be in 64-bit addressing mode (AMODE 64). The switch to AMODE 64 is done with the SAM64 instruction. The z/Architecture provides three new Set Addressing Mode (SAM) assembler instructions that allow you to change the addressing mode. These instructions are SAM24, SAM31, and SAM64, which change the current addressing mode to AMODE 24, AMODE 31, and AMODE 64, respectively. SeeTable 2-3 for a summary of the Set Addressing Mode instruction, and the resulting mode bit settings in the Program Status Word (PSW). There are also many new assembler instructions in the z/Architecture to manipulate data above the bar and in the 64-bit registers.

Table 2-3 Set Addressing Mode instructions

Instruction	HEX	PSW b.31	PSW b.32	AMODE
SAM24	'010C'	0	0	24
SAM31	'010D'	0	1	31
SAM64	'010E'	1	1	64

2.2.2 IBM middleware support

As z/OS V1R2 introduces the basic 64-bit virtual storage support in the operating system, middleware will be able to take advantage of this support at a later stage. In particular, products like DB2 and WebSphere will, in the future, get benefits derived from the large amount of virtual storage for data made available by the operating system.

DB2

DB2 will be one of the first subsystems to take advantage of 64-bit data addressability. It has superb abilities to handle varied large system workloads efficiently, including transaction and large query environments.

With 64-bit virtual storage support, DB2 will relieve virtual storage constraints. There will be enhancements in database buffer management support to provide continued growth in the large systems transaction environment. All existing 31-bit DB2 applications will benefit transparently from the local availability of data made available through 64-bit virtual storage support.

WebSphere

WebSphere Application Server for z/OS is the strategic server for e-business. It supports component-based program development through the Enterprise Java Beans (EJB) programming model.

With 64-bit virtual storage support in the Java Virtual Machine (JVM), the z/OS Websphere Application Server will be able to support a large number of EJBs and a large number of data objects, in particular very large data objects.

2.2.3 IARV64 system service

The IARV64 macro allows a program to allocate and use the full range of virtual storage in a 64-bit address space. The macro allocates storage areas above the 2 GB address, above the bar, in entities called *memory objects*. It also frees memory objects and can manage the physical storage frames used to back up the virtual storage. Each memory object is a multiple of one megabyte in size, which is the segment size, and begins on a megabyte boundary. All the memory management operations must be performed within a memory object, no operation can cross a memory object boundary.

You can think of the IARV64 macro as the GETMAIN, FREEMAIN, PGSER, or STORAGE macro for use with virtual storage above the bar. The GETMAIN, FREEMAN, STORAGE, and CPOOL macros do not allocate storage above the 2-gigabyte address, nor do callable cell pool services.

SYSSTATE macro

Before using the IARV64 service, issue the SYSSTATE ARCHLVL=2 macro, so that the IARV64 generates the correct 64-bit parameter addresses.

You may have used the SYSSTATE macro for setting a global variable to indicate whether your program is in primary or access register address space control mode.

You can also use SYSSTATE to set global variables that tell the architecture level of your system, and if your program is running in 64-bit addressing mode.

Set architecture level

Use the ARCHLVL= parameter on the macro to indicate the architecture level of your system by setting the global symbol &SYSALVL. The level can be 0, 1, or 2. These values have the following meanings:

0 The architecture is ESA/390.

1 The architecture is ESA/390, but includes the ESA/390 architecture items required by OS/390 V2 Release 10 (for example, the relative/immediate instructions).

2 The architecture is z/Architecture. Macros that pay attention to &SYSALVL global symbol will avoid generating z/Architecture instructions when an ARCHLVL less than 2 is in effect.

Set addressing mode

You can use the AMODE= parameter on the macro to indicate whether your program is in AMODE 64 or not by setting the global variable &SYSAM64.

AMODE64=NO should be specified for programs or parts of programs that do not run in 64-bit addressing mode.

AMODE64=YES should be specified for any part of your program that runs in 64-bit addressing mode. AMODE64=YES builds a user parameter list consisting of 8–byte entries for macros capable of running in z/Architecture mode, such as LINK, LOAD, XCTL, and ATTACH.

2.2.4 IARV64 service requests

There are multiple service requests available with the IARV64 macro, including the following:

GETSTOR Create a memory object.

DETACH Free one or more memory objects. The memory object is freed.

DISCARDDATA Discard data within physical pages of one or more memory objects.

PAGEIN Give notification that data within physical pages of one or more memory objects are needed in the near future.

PAGEOUT Give notification that data within physical pages of one or more memory objects will not be used in the near future.

CHANGEGUARD Requests that a specified number of application pages be changed from the guard state to the usable state or vice versa.

PAGEFIX Fix physical pages within one or more memory objects.

PAGEUNFIX Unfix physical pages within one or more memory objects.

LIST Request a list of memory objects.

The format of the macro instruction is:

```
name IARV64 REQUEST=service request,options...
```

See *z/OS MVS Assembler Services Reference*, SA22-7606 for a complete write-up of the macro and its parameters. Invalid requests get a system abend DC2.

GETSTOR service

Use the GETSTOR service to create a memory object. This service creates a memory object with the characteristics defined by your program. The memory object can also contain a guard area if you specify so. The address of the lowest virtual address of the memory object is returned from the service to your program. You can think of this address as the name of the memory object. If you specified a guard area staring at the low virtual address, the address of the guard area will be returned. This area is not usable by your program.

You can specify whether or not the data in the memory object is to be included in the dump when a dump is requested with SDATA=(RGN) by coding the SVCDUMPRGN=YES/NO parameter.

You should use this parameter with caution. Remember that no control information or system control blocks reside above the bar, it is only user data. Be aware that the default is “yes,” and depending on the size of the memory object, it may take a long time to dump the data.

You can specify an 8-byte user token, which is used to group memory objects together for later detach or list service requests.

```
IARV64 REQUEST=GETSTOR,SEGMENTS=NUMSEGS,SVCDUMPRGN=NO, ....
```

Figure 2-8 GETSTOR service

DETACH service

Use DETACH service to free one or more memory objects as specified by your program. The physical frames that back the virtual storage occupied by the memory object are returned to the system. You can delete a single memory object, or you can pass a user token that identifies the related memory objects that you want to delete in this operation. The memory objects not associated with this user token are not affected.

```
IARV64 REQUEST=DETACH,USERTKN=U_TOKEN, .....
```

Figure 2-9 DETACH service

DISCARDDATA service

Use the DISCARDDATA service to notify the memory manager that the data in certain pages is no longer needed, and that the physical resources backing these pages can be freed.

You must provide a list of page ranges. The ranges consist of 1 to 16 pairs of 8-byte virtual addresses and 8-byte counts of 4 KB pages. Each range must be within the same memory object, and each virtual address must be on a page boundary.

PAGEFIX service

Use the PAGEFIX service to fix frames in real storage that back one or more memory objects. The memory objects can be in primary or home address space. The service makes frames resident in real storage, thus they are not candidates for page stealing. You can also inform the memory manager that the page fix is to be of long duration. A few seconds is considered a long time. Your program must be authorized, supervisor state or Key 0 - 7, to use this service.

PAGEUNFIX service

Use the PAGEUNFIX service to undo the services you requested with PAGEFIX. Your program must also be authorized to use this service.

PAGEIN service

Use the PAGEIN service to notify the memory manager that the data in certain pages will be referenced soon, and it would be good if these pages were paged in if they are not already backed by real storage.

You must provide a list of page ranges. The ranges consist of 1 to 16 pairs of 8-byte virtual addresses and 8-byte counts of 4 KB pages. Each range must be within the same memory object, and each virtual address must be on a page boundary.

The PAGEIN service is just a suggestion to the system. It may not be done if the paging subsystem is too busy.

PAGEOUT service

Use the PAGEOUT service to notify the memory manager that the data in certain pages will not be used for some time, and that these pages are a good candidate for paging out of real storage. “Some time” here is measured in seconds or longer. Pages that are already fixed in real storage are unaffected by this request.

You must provide a list of page ranges. The ranges consist of 1 to 16 pairs of 8-byte virtual addresses and 8-byte counts of 4 KB pages. Each range must be within the same memory object, and each virtual address must be on a page boundary.

The PAGEOUT service is just a suggestion to the system. It only makes frames look old, they may not be stolen if there is no demand for real storage. This service is no guarantee that the pages are written to auxiliary storage.

CHANGEGUARD service

Use the CHANGEGUARD service to change the size of the guard area and the usable area of a memory object. The memory object is named by the virtual storage address which was returned from the GETSTOR service. You can specify that an amount in megabyte chunks is to be changed from guard area to usable area, and vice versa.

LIST service

Use the LIST service to request information about the usable areas of memory objects. The information is returned in a work area you specify. The service will return one element in the work area array for each memory object in the address space. The information in the element contains the storage key of the memory object, its lowest usable virtual address, and its highest usable virtual address.

Range lists services

Several services use a list of virtual addresses, a range list. Such a list can contain from 1 to 16 ranges. The number of ranges in the list is specified by the RANGELIST= parameter to the actual service. An entry in the range list is 16 bytes long. The range list consist of pairs of 8-byte virtual addresses and 8-byte counts of 4 KB pages, this is illustrated in Figure 2-10. All virtual addresses must be on a 4 K boundary. The starting virtual address and the ending virtual address must be within a single memory object. The ending address is calculated from the number of 4 KB pages specified.

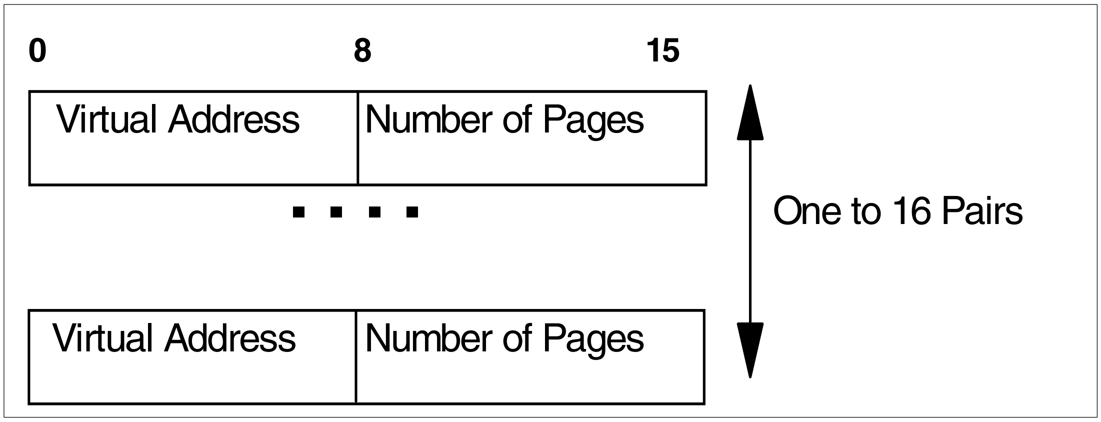


Figure 2-10 Range list format

IARV64 error codes

IARV64 has four return codes. The return codes and their meanings are:

00 (0)	Successful completion.
04 (4)	Successful completion, with exception.
08 (8)	The request is rejected due to non-system failure.
0C (12)	The request is rejected due to system failure.

There is a new abend code DC2 associated with IARV64, meaning that IARV64 services failed due to an invalid request.

2.3 Using virtual storage above 2 GB

When designing your application you now have the possibility to use a single memory model, with virtual storage above the bar. The virtual storage in the address space is 16 exabyte (16 EB) in size.

Before z/OS V1R2, a program’s need for virtual storage in excess of what the former 2 GB address space provided was accomplished by creating one or more data spaces or hiperspaces, and then using some sort of storage management schema to keep track of data in those spaces. With the 16 EB address space, this complex programming belongs to the past. A program can now simply have “as much” virtual storage as it needs, with the data contained in its home address space. This is an advantage for programs that, for example, use very large buffers or handle sparse arrays.

A program does not start in 64-bit addressing mode (AMODE 64). All programs start in 24- or 31-bit addressing mode (AMODE 24 or AMODE31). To use storage above the bar, the program must be in 64-bit addressing mode (AMODE 64). The program changes its addressing mode to AMODE 64 with the SAM64 assembler instruction. Then it must obtain a memory object through the IARV64 assembler macro. When the program is in AMODE 64, it continues to run below the bar, but it can access data in memory objects above the bar.

Most macros that can be used in 64-bit addressing mode require prior invocation of the SYSSTATE macro with ARCHLVL=2 option specified. This is necessary for the macro to generate correct code sequences.

There are new assembler instructions for manipulating data above the bar and in the 64-bit registers.

2.3.1 Program example obtaining storage

Example 2-1 shows an assembler program that obtains a 4 MB memory object. The program then writes some data into every 4 KB page of that memory object, and then deletes it.

Example 2-1 Program using virtual storage above the bar

```
         TITLE 'TEST V64'
V64      CSECT
V64      AMODE 31
V64      RMODE 31
*  ENSURE CORRECT MACRO EXPANSION FOR 64 BIT ADDRESSING
         SYSSTATE ARCHLVL=2
*  BEGIN ENTRY LINKAGE AND GET SAVEAREA
         BAKR  14,0
         CNOP  0,4
         BRAS  12,@PROG
         DC    A(@PROG)
@PROG    LLGF  12,0(12)
         USING @PROG,12
         LHI   0,AREAL
         STORAGE  OBTAIN,LENGTH=(0),SP=0,CALLRKY=YES
         LLGTR 13,1
         USING @AREA,13
         MVC   4(4,13),=C'ITSO'         EYECATCHER
*  END ENTRY LINKAGE
*
*  CHANGE TO AMODE 64
         SAM64
*  GET VIRTUAL STORAGE ABOVE THE BAR
         IARV64 REQUEST=GETSTOR,                                       C
               SEGMENTS=MO_SIZE,                                       C
               USERTKN=U_TOKEN,                                        C
               ORIGIN=V64_ADDR
         LTGR  15,15                    GOT MEMORY OBJECT ?
         BC    8,WG                     - YES, OK
         DC    H'0'                     - NO, INVESTIGATE
*  START WORK WITH DATA IN STORAGE ABOVE THE BAR
WG       WTO   'GOT V64',ROUTCDE=11
         LG    4,V64_ADDR               GET ADDRESS OF MEMORY OBJECT
         LHI   2,256*4                  LOOP COUNTER, TOUCH ALL PAGES
TOUCH    MVC   0(L'DATA,4),DATA         MOVE IN SOME DATA
         AHI   4,4096                   TO NEXT PAGE
         BRCT  2,TOUCH                  LOOP BACK AND TOUCH NEXT PAGE
*  DETACH VIRTUAL STORAGE ABOVE THE BAR
         IARV64 REQUEST=DETACH,                                        C
```

```
               MATCH=USERTOKEN,                                    C
               USERTKN=U_TOKEN,                                    C
               COND=YES
         LTGR  15,15                FREED MEMORY OBJECT ?
         BC    8,WD                 - YES, OK
         DC    H'0'                 - NO, INVESTIGATE
WD       WTO   'DETACHED V64',ROUTCDE=11
*  BEGIN EXIT LINKAGE
         LHI   0,AREAL
         LR    1,13
         STORAGE  RELEASE,LENGTH=(0),ADDR=(1),SP=0,CALLRKY=YES
         PR
*  END EXIT LINKAGE
@DATA    DS    0D
MO_SIZE  DC    FD'4'                MEMORY OBJECT IS 4 MB
U_TOKEN  DC    FD'1'
DATA     DC    C'DATA ABOVE THE BAR'
         LTORG
@AREA    DSECT
SAVEAREA DS    36F
V64_ADDR DS    AD                   'NAME' OF MY MEMORY OBJECT
AREAL    EQU   *-@AREA
         END   V64
```

2.4 Controlling storage usage

Your installation can limit the maximum physical memory resources, both real storage and auxiliary storage (paging space) that can be committed by a job. We recommend that you do this.

The enormous amount of virtual storage a job can obtain and manipulate can put extreme load on your backing storage, real storage frames, and auxiliary storage slots on the paging data sets. Before you start using virtual storage above the bar, try to estimate how much it will be used, and make sure you have enough real storage and paging space to back it up. Otherwise, you may experience extreme paging, which may affect the overall performance of your system.

There are several ways to limit the resource usage for virtual storage above the 2 GB virtual address:

- Through JCL on the specific job with the new MEMLIMIT JCL keyword
- Through an installation-defined default value in SMF parameters, SMFPRMxx in parmlib
- Through the SMF installation exit IEFUSI

If you do not set any MEMLIMIT value, the system uses the default value, which is 0. This means that no address space can use virtual storage above the bar. If you want to use virtual storage above the bar, the MEMLIMIT value must be explicitly set.

2.4.1 MEMLIMIT and JCL

For virtual storage below 2 GB you may limit the size of virtual storage a job can obtain through the REGION keyword on the JOB or EXEC Job Control Language (JCL) statements, and control and override this through the SMF exit IEFUSI. A value equal to 0K or 0M gives the job all the storage available below and above the 16 megabyte line.

For virtual storage above the bar, a new JCL keyword, MEMLIMIT is introduced on the JOB and EXEC JCL statements. For virtual storage above the bar, there is no practical limit to the amount of virtual address range an address space can request. However, there are practical limits to the real storage and auxiliary storage needed to back the request. Therefore, a limit is placed on the amount of usable virtual storage above the bar an address space can use at any one time.

> **Important:** MEMLIMIT controls the amount of usable storage above the 2 GB line.

Job statement

You can use the MEMLIMIT parameter to specify the limit on the total number of usable virtual storage pages above the bar for a single address space.

The syntax of the MEMLIMIT parameter is:

```
MEMLIMIT=nnnnnP
```

where nnnnn can be any number from 0 - 99999 and specifies the limit on total usable virtual storage pages above the bar in the single address space. The value may be expressed in megabytes (M), gigabytes (G), terabytes (T), or petabytes (P).

You can also use:

```
MEMLIMIT=NOLIMIT
```

This specifies that there is no limit to the virtual storage pages to be used above the bar.

If you specify MEMLIMIT on the JOB statement, this value overrides MEMLIMIT on the EXEC statement.

If MEMLIMIT is not specified, SMF provides a default value, except when REGION=0K/0M is specified, in which case the default is NOLIMIT.

The IEFUSI SMF installation exit can override any JCL- or SMF-supplied value.

The job statement in Figure 2-11 specifies that the job is limited to the use of 20 gigabytes of usable virtual pages above the bar.

|

```
//RFR01   JOB (999,POK),'RF',CLASS=A,MSGCLASS=U,NOTIFY=&SYSUID,
// REGION=0M,MEMLIMIT=20G
```

Figure 2-11 MEMLIMIT on a JOB statement

Job statement examples

The specifications of MEMLIMIT on the job card can be done as follows:

```
//TC1 JOB  MEMLIMIT=50G,REGION=0M
//TC2 JOB  MEMLIMIT=125M,TIME=NOLIMIT
//TC3 JOB  MEMLIMIT= 9T,MSGLEVEL=1
//TC4 JOB  REGION=3M,MEMLIMIT=16384P
//TC5 JOB  REGION=125M,MSGLEVEL=(1,1),MEMLIMIT=NOLIMIT,MSGCLASS=A
```

Figure 2-12 JOB statement MEMLIMIT examples

EXEC statement

You can use the MEMLIMIT parameter to specify the limit on the total number of usable virtual storage pages above the bar for a single address space.

The syntax of the MEMLIMIT parameter is:

```
MEMLIMIT=nnnnnP
```

where nnnnn can be any number from 0 - 99999 and specifies the limit on total usable virtual storage pages above the bar in the single address space. The value may be expressed in megabytes (M), gigabytes (G), terabytes (T), or petabytes (P).

You can also use:

```
MEMLIMIT=NOLIMIT
```

This specifies that there is no limit to the virtual storage pages to be used above the bar.

The JOB statement MEMLIMIT parameter applies to all steps of the job and overrides any EXEC statement MEMLIMIT parameter. If MEMLIMIT is not specified on the EXEC statement, SMF provides a default value, except when REGION=0K/0M is specified, in which case the default is NOLIMIT. The IEFUSI installation exit can override any JCL- or SMF-supplied value.

The EXEC statement in Figure 2-13 specifies that this step is limited to the use of 10000 megabytes of usable virtual pages above the bar, depending on the JOB card or the IEFUSI installation exit override.

EXEC statement examples

The specifications of MEMLIMIT on the EXEC statement can be done as follows:

```
//STEP1 EXEC PGM=V64,MEMLIMIT=10000M
//STEP1 EXEC  PGM=TST6,MEMLIMIT=6400M
//STEP1 EXEC PGM=TST7,MEMLIMIT=3P...
//STEP1 EXEC MYPROC,MEMLIMIT=NOLIMIT...
```

Figure 2-13   MEMLIMIT on a EXEC statement

2.4.2 MEMLIMIT and ESA/390 mode

If you submit JCL with the MEMLIMIT keyword to a z/OS V1R2 system for JCL conversion, and that system is in ARCHLVL =1 (ESA/390 mode) the job will fail with message IEF897I as shown in Figure 2-15 on page 30 for the JOB statement, and Figure 2-16 on page 30 for the EXEC statement. In order to determine the architecture level of the system use the **D IPLINFO** command, as shown in Figure 2-14.

```
D IPLINFO
IEE254I  13.45.27 IPLINFO DISPLAY 128
  SYSTEM IPLED AT 06.38.48 ON 05/27/2001
  RELEASE z/OS   01.02.00
  USED LOAD59 IN SYS0.IPLPARM ON 3800
  ARCHLVL = 1   MTLSHARE = N
  IEASYM LIST = XX
  IEASYS LIST = (00,00) (OP)
  IODF DEVICE 3800
  IPL DEVICE 3833 VOLUME Z02RA1
```

Figure 2-14   Display IPLINFO for an ESA/390 system

```
IEF897I ESA MODE SYSTEM CANNOT PROCESS MEMLIMIT KEYWORD ON THE JOB STATEMENT
```

Figure 2-15 IEF897I for the JOB statement

```
IEF897I ESA MODE SYSTEM CANNOT PROCESS MEMLIMIT KEYWORD ON THE EXEC STATEMENT
```

Figure 2-16 IEF897I for the EXEC statement

If you submit your JCL with the MEMLIMIT keyword to a pre z/OS V1R2 system for JCL, or the JCL is converted by a pre z/OS V1R2 system in the sysplex, you will get a JCL error like that shown in Figure 2-17.

```
IEFC630I UNIDENTIFIED KEYWORD MEMLIMIT
```

Figure 2-17 MEMLIMIT JCL error

2.4.3 Installation default MEMLIMIT

The SMFPRMxx member allows you to control how System Management Facilities (SMF) works at your installation. To limit the use of real and auxiliary storage an address space can use for memory objects above the bar, you can set an installation-defined MEMLIMIT value.

A new parameter, MEMLIMIT, is added to the SMFPRMxx parmlib member in z/OS V1R2. MEMLIMIT specifies the default value that will be used by jobs that do not have a MEMLIMIT value explicitly provided on the JOB or EXEC statement, or by jobs that have specified REGION=0K/M on the JOB or EXEC statement.

MEMLIMIT specified in the SMF installation exit IEFUSI overrides all other sources of MEMLIMIT.

The SMFPRMxx parameter can be specified as:

```
   MEMLIMIT=NOLIMIT
```

This means that there is no limit to the use of virtual storage above the bar for a single address space.

If you want to set an installation default limit you must specify:

```
   MEMLIMIT=nnnnnP
```

where nnnnn can be any number from 0 - 99999 and specifies the limit on total usable virtual storage pages above the bar in the single address space. The value may be expressed in megabytes (M), gigabytes (G), terabytes (T), or petabytes (P).

The default value for the SMFPRMxx parameter MEMLIMIT is 0M. This means that no address space can use virtual storage above the bar. However, if REGION=0K or 0M is specified on the JOB or EXEC card, the default value is NOLIMIT.

For example, to set 1500 gigabytes as the installation default value, specify MEMLIMIT=1500G, or to set 15 exabytes as the default, specify MEMLIMIT=15000P.

The operator console command **D SMF,O** displays the current MEMLIMIT. See Figure 2-18 on page 31 for an example of the **D SMF,O** command and its response.

```
D SMF,O
IEE967I 14.33.11 SMF PARAMETERS 604
        MEMBER = SMFPRM00
        MEMLIMIT(00000M) -- DEFAULT
        SYNCVAL(00) -- DEFAULT
        DUMPABND(RETRY) -- DEFAULT
        SUBSYS(STC,NOINTERVAL) -- SYS
        SUBSYS(STC,NODETAIL) -- SYS
```

Figure 2-18 Display SMF,O

Determine MEMLIMIT for a jobstep

Figure 2-19 describes what value of MEMLIMIT is passed to user exit IEFUSI.

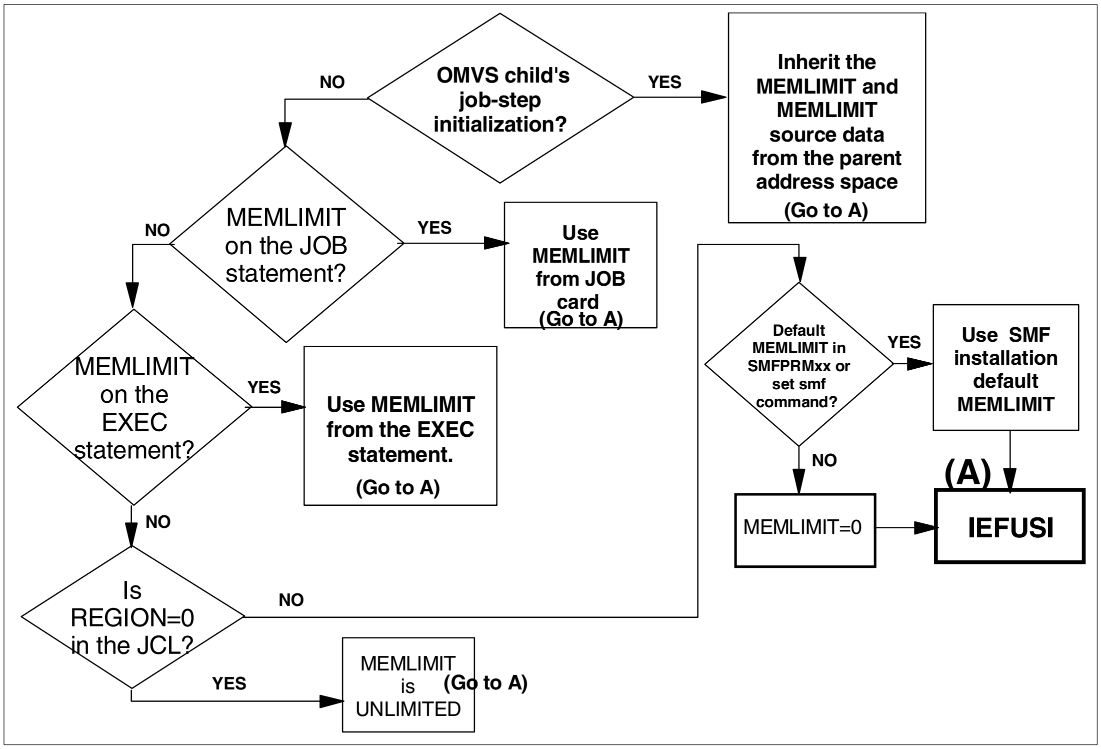

OMVS child's job-step initialization?
NO
YES
Inherit the MEMLIMIT and MEMLIMIT source data from the parent address space
(Go to A)
MEMLIMIT on the JOB statement?
NO
YES
Use MEMLIMIT from JOB card
(Go to A)
MEMLIMIT on the EXEC statement?
YES
Use MEMLIMIT from the EXEC statement.
(Go to A)
NO
Is REGION=0 in the JCL?
NO
YES
MEMLIMIT is UNLIMITED
(Go to A)
Default MEMLIMIT in SMFPRMxx or set smf command?
YES
Use SMF installation default MEMLIMIT
NO
MEMLIMIT=0
(A)
IEFUSI

Figure 2-19 Determine the MEMLIMIT for a jobstep

2.4.4 Installation SMF exit IEFUSI

IEFUSI is an SMF exit that receives control before each step is initiated. A return code from this exit tells the system whether the job step can start or not. The exit is used to validate, among other things, the region size for the programs that run under this job step.

You can use this exit to limit the use of the 16 EB address space above the bar. You can control and override all specifications of the MEMLIMIT parameter given in JCL or the SMFPRMxx PARMLIB member.

As with other SMF exits, you get an address to a parameter list passed in general register 1. The parameter list passed to IEFUSI consist of a list of addresses. Word 9 of this address list, the 64-bit flagword, is where you can find information on the job step's MEMLIMIT value.

A return code of 4 from this exit cancels the job step.

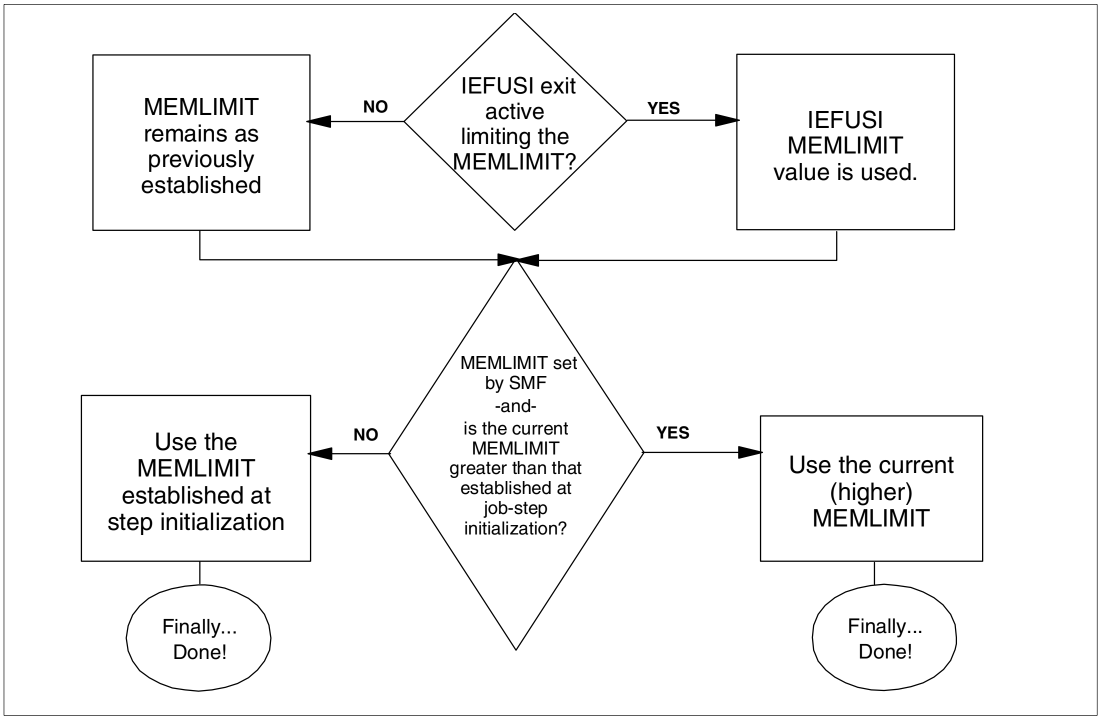

IEFUSI exit active limiting the MEMLIMIT?
NO
YES
MEMLIMIT remains as previously established
IEFUSI MEMLIMIT value is used.
MEMLIMIT set by SMF -and- is the current MEMLIMIT greater than that established at job-step initialization?
NO
YES
Use the MEMLIMIT established at step initialization
Use the current (higher) MEMLIMIT
Finally... Done!
Finally... Done!

Figure 2-20 Determine MEMLIMIT in IEFUSI exit

2.4.5 RMF

One of the RMF records, SMF record type 79, subtype 1, is updated to include the MEMLIMIT and the source of MEMLIMIT setting for an address space.

2.5 I/O operations

The EXCP system service macro can be used to do input/output (I/O) operations referencing data in virtual storage above the bar.

In order to do this I/O operation, you need to use 64-bit virtual indirect address words (IDAW) in your channel programs. 64-bit virtual IDAWs are only supported in ARCHLVL 2 mode, and only by devices that support 64-bit real IDAWs. Currently this is only disk and tape.

To use 64-bit virtual IDAWs, your program must create an Input/Output block Extension (IOBE), and the IOBEEIDA flag, X’10’, in IOBEFLG2 must be set to indicate the use of 8-byte IDAWs.

31-bit and 64-bit IDAWs cannot be intermixed in the same channel program.

An abend with system completion code 800 will be issued if you request 64-bit IDAWs during your EXCP processing and they are not supported. The system completion code 800 means that during processing of an I/O request, execute channel program (EXCP) processing encountered an error. Each system 800 abend code has a reason code that explains more about the error. In this case you will get a reason code of 1, which indicates that an error occurred during indirect address word (IDAW) or channel command word (CCW) validity check processing.

2.6 Dumping virtual storage above 2GB

The SVC Dump service has been enhanced for the 64-bit virtual address space support through the SDUMPX macro. This is done with the LIST64, SUMLIST64, and enhancements to the SDATA=RGN parameters of the SDUMPX macro.

Dumps taken to SYSMDUMP data sets will be enhanced for 64-bit virtual support. Dumps taken to SYSABEND or SYSUDUMP data sets has not been enhanced in any way for 64-bit virtual support.

To limit the size of an SVC dump, we recommend that you use the SDATA=RGN parameter on the SDUMPX macro with caution. This is when you are in AMODE 64 and have active memory objects created with the IARV64 REQUEST=OBTAIN and you have used the default operand SVCDUMPRGN=YES on this macro. Data above the bar is user data containing no system control information, and is probably of little diagnostic value.

2.6.1 SDUMPX Macro

The SDUMPX macro invokes SVC dump to provide a fast unformatted dump of virtual storage or coupling facility structure information to a data set. SDUMPX is similar to SDUMP, except that SDUMPX can generate code and addresses that are appropriate for AR mode, whereas SDUMP cannot.

To dump data space storage, issue SDUMPX and specify one of the following parameters: DSPLIST, LISTD, LIST64, SUMLSTL or SUMLIST64.

The SDUMP macro can be invoked in AMODE 64. To run in 64-bit addressing mode, issue the SYSSTATE AMODE64 variable set to YES prior to invoking SDUMPX.

The SDATA=RGN, and LIST64, and SUMLIST64 parameters apply for memory objects above the bar.

SDATA=RGN

When the SDATA=RGN parameter is specified, the SVC Dump for a given address space being dumped contains all of the virtual storage areas it captured, as defined prior to this z/OS V1R2.

For z/OS V1R2 environments, SVC Dump additionally dumps the virtual storage for 64-bit addressable memory objects that were created with the default SVCDUMPRGN=YES attribute, as shown in Figure 2-21.

```
IARV64 REQUEST=GETSTOR,...SVCDUMPRGN=YES,...
```

Figure 2-21 GETSTOR with SVCDUMPRGN

Memory objects that are created with the SVCDUMPRGN=NO attribute are not included in an SVC dump when SDATA=RGN is specified. Ranges within such memory objects can be captured within an SVC Dump using LIST64 and SUMLIST64.

The parameter list and all non-register notation keyword address parameters must reside in 31-bit addressable virtual storage, including for invokers that are executing in 64-bit addressing mode.

LIST64= listaddr

The listaddr operand specifies a storage location describing a list of STOKEN-qualified virtual storage address ranges to be included in the dump. The address ranges are described with 64-bit starting and ending addresses and may refer to any address space virtual storage, including memory objects in virtual storage above the bar. Each starting address must be lower than its corresponding ending address.

The LIST64 parameter must address a storage location residing in 31-bit addressable virtual storage, but the list content at that storage location may, in turn, address 64-bit addressable virtual storage.

The LIST64 parameter is mutually exclusive with the LISTD parameter.

SUMLIST64= listaddr

The listaddr operand specifies a storage location describing a list of virtual storage address ranges, qualified by STOKEN or ALET, to be included in a summary dump. The address ranges are described with 64-bit starting and ending addresses and may refer to any address space virtual storage, including memory objects in virtual storage above the bar.

The SUMLIST64 parameter is mutually exclusive with the SUMLSTL parameter.

2.7 z/Architecture considerations

The z/Architecture definition separates the control of 64-bit addressing mode from 64-bit operations. This style of definition allows software to be extended to 64-bit in an evolutionary manner. 32-bit and 64-bit operations are supported in 24-, 31- and 64-bit addressing mode. Any program can switch between addressing modes and intermix 32-bit and 64-bit operations.

The z/Architecture is a 64-bit architecture with several types of 64-bit registers and areas, such as 64-bit general purpose registers (GPR). The numbering of the bits for these entities is changed to be 0 to 63. This may cause confusion when dealing with some of the compatible aspects of the architecture. For example, the ESA/390 Load instructions set bits 0-31 of a general purpose register. In z/Architecture, the Load instructions set bits 32-63, due to the renumbering of the bits. To ease understanding, sometimes the ESA/390 bit positions will also be provided in braces. Bit positions are usually designated after a dot. For example:

```
LR R1,R2 sets R1 .32-63{0-31} from R2 .32-63{0-31}
```

2.7.1 Addressing modes

There are 3 new instructions which change addressing mode without branching:

- Set Addressing Mode to 24-bit (SAM24)
- Set Addressing Mode to 31-bit (SAM31)
- Set Addressing Mode to 64-bit (SAM64)

There are 2 instructions which change addressing mode and branch:

- Branch and Save and Set Mode (BASSM)
- Branch and Set Mode (BSM)

Since z/Architecture allows the intermixing of addressing modes and operand precision, care is taken to ensure that old 31-bit programs could not inadvertently destroy register information which new 64-bit programs assumed to be preserved.

When executing in 24- or 31-bit addressing mode and issuing only instructions which are defined in ESA/390, the high-order 32-bit of the 64-bit general purpose registers are opaque. Therefore, “old” programs, which do not save/restore the full 64-bit registers, cannot unknowingly alter the contents of the high-order halves of the registers.

The high-order halves of the registers are visible only in 64-bit mode or when executing new instructions.

2.7.2 Modal instructions

Certain instructions leave bits 0-31 of a general register unchanged in the 24-bit or 31-bit addressing mode, but place or update address or length information in them in the 64-bit addressing mode. These instructions are sometimes called modal instructions.

Modal instructions operate differently depending upon the addressing mode. Usually the difference is the width of register operands. Examples of modal instructions are:

```
LA, MVCL, TRT, BASR, BASSM
```

Instructions like BCR are not considered modal because address generation naturally truncates the address taken from the register according to current addressing mode.

2.7.3 Non-modal instructions

The term non-modal applies to instructions which perform the same operation regardless of addressing mode. For these instructions, the addressing mode is only used during storage operand address generation. For example:

```
L always loads R1 .32-63{0-31}  from D2(X2,B2) .0-31
```

Non-modal 32-bit operand instructions behave as in ESA/390, regardless of addressing mode. Bits 0-31 of the 64-bit GPRs are unexamined and unmodified. Examples are:

```
LR, AR, ALR, L, A, AL
```

The operation is the same in all addressing modes.

2.7.4 64-bit registers

The GPR is treated as 64-bits for:

- Address generation in 64-bit mode
- GPR operands of non-modal 64-bit instructions
- GPR operands of modal instructions in 64-bit mode

The GPR is treated as 32-bits for:

- Address generation in 24/31-bit modes (so it seems)
- GPR operands of non-modal 32-bit instructions
- GPR operands of modal instructions in 24/31-bit modes

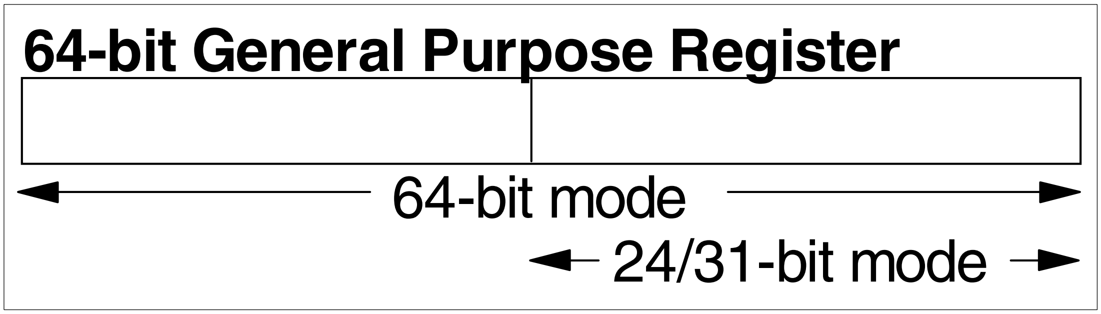


Figure 2-22 64-bit general purpose register

2.8 Restrictions

WIth z/OS Version 1 Release 2, there are some restrictions that apply to the 64-bit virtual implementation as follows:

- No expanded storage.
- ADMF ==> Fast Sync'h Data Mover Fac. FSDMF. Docs for IOSADMF
- Only data—and not programs—can use storage above the bar.
- Data spaces remain at 31-bit addressing. There is no support for data spaces larger that 2 GB.
- Hiperspaces larger than 2 GB are not supported.
- Data In Virtual (DIV) support is not extended to use virtual storage above the bar.
- Virtual I/O (VIO) support is not extended to use virtual storage above the bar.
- Copy-on Write is not supported for virtual storage above the bar.
- Change Key is not supported for virtual storage above the bar.

2.8.1 Subspaces

Subspace capacity is not extended to virtual storage above the bar.

If your program attempts to get a memory object above the bar in an address space that currently or previously obtained a subspace, it will fail with a system abend DC2.

On the other hand, if you attempt to get a subspace in an address space that either currently or previously has had memory objects above the bar, this attempt will fail with a system abend code 3C6.

3

z/OS base control program changes

This chapter describes changes made to the base control program (BCP):

- Unconditional log close
- Open data set limits
- Service Aids
- End-of-memory resource manager
- Global resource serialization wildcard support

3.1 Unconditional log close

The SYSLOG data set is allocated by the **WRITELOG** command and is maintained on the JES spool. When a spool volume needs to be taken offline, that spool volume cannot go into a removable status because the SYSLOG data set is allocated by WRITELOG.

WRITELOG,CLOSE does not work because the command does not permit closure of the SYSLOG data set if this is the only HARDCOPY log. Therefore, prior to z/OS Version 1 Release 2, it was necessary to schedule an MVS IPL to free up the SYSLOG data set.

3.1.1 Hardcopy log enhancements

z/OS Version 1 Release 2 introduces the ability to allow a hardcopy log to be closed without having a backup hardcopy log. This change is introduced by the following new support:

- A new parameter on the **VARY HARDCPY** command
- Changes to messages IEE012A and IEE299I
- An enhancement to the **D C,HC** command

Hardcopy command changes

The SYSLOG must have been active before you can issue the following command with the new parameter, UNCOND:

```
VARY SYSLOG,HARDCPY,OFF,UNCOND
```

The hardcopy log must have been started for the command to work. The hardcopy log buffers cannot be saved once this command has been issued.

LOGLIM parameter

A write-to operator queue element (WQE) is created for every WTO/WTOR request. The LOGLIM parameter specifies the maximum number of buffers that the system can use to process write-to-log (WTL) messages. Each buffer is 140 bytes, and is obtained from the extended common service area (ECSA). These log buffers are only created when SYSLOG is the logging mechanism. Therefore, they can only be saved when SYSLOG is turned off.

Turning off OPERLOG or a printer is allowed, but should be very rare. It is provided for cases where an installation has its own completely separate log.

Note: JES3 systems using the DLOG require a higher default for the LOGLIM parameter to prevent WTL buffer shortages and SYSLOG constraints. The IBM recommendation is to set the LOGLIM value based on the JES3 configuration at your installation, at least equal to:

```
LOGLIM = (4000 + (2000 x the-number-of-JES3-locals))
```

Message IEE299I

This message is issued when a **VARY SYSLOG,HARDCPY,OFF** command requested that the hardcopy log be removed from a device or that the operations log be deactivated. However, processing the command would have resulted in neither the hardcopy log, the operations log, nor a hardcopy device remaining active at a time when the hardcopy function is required. The changes to the message are as follows:

```
IEE299I SYSLOG REQ’D FOR HARDCPY - USE UNCOND TO TURN OFF HARDCOPY
```

Previously, IEE299I was simply:

```
IEE299I DEVICE REQ'D FOR HARDCPY
```

The new text is added to remind the user about the UNCOND parameter.

If it is absolutely necessary to turn off the hardcopy function in this release, issue:

```
VARY SYSLOG,HARDCPY,OFF,UNCOND
```

This should be done temporarily, and only as a last resort, to repair hardcopy functions.

Message IEE012A

When you issue a **VARY SYSLOG,HARDCPY,OFF,UNCOND** command, MVS saves messages to be hardcopied later. In the meantime, if the limit of LOGLIM has been reached, messages issued after this point are not hardcopied, unless LOGLIM is increased or hardcopy is re-activated.

When the LOGLIM limit is reached, the following message is issued:

```
IEE012A NO LONGER SAVING MESSAGES FOR HARDCOPY, LOGLIM REACHED
```

The operator response is to either activate a hardcopy medium, or increase the value of LOGLIM with the following command:

```
K M,LOGLIM
```

Display console command

You can display the log buffer limit, and the number of buffers in use, by issuing the **D C,HC** command, as shown in Figure 3-1.

```
VARY SYSLOG,HARDCPY,OFF,UNCOND
IEE338I SYSLOG   INACTIVE AS HARDCPY
D C,HC
IEE889I 14.36.37 CONSOLE DISPLAY 425
MSG: CURR=2    LIM=1500 RPLY:CURR=2    LIM=999  SYS=SC59
HARDCOPY SUSPENDED ON THIS SYSTEM
LOG BUFFERS IN USE:        2  LOG BUFFER LIMIT:     6000
```

Figure 3-1 Changed commands for unconditional log close

3.1.2 Usage example

The following example shows how to use the new parameter when it is required to repair the SYSLOG.

1. Prepare to save as many messages as possible. These buffers are 140 bytes each and allocated in common storage (ECSA), so that specifying a large number should be temporary.

   ```
   K M,LOGLIM=999999
   ```

2. Enter the following command to eliminate the hardcopy medium. Messages are being saved in the log buffers.

   ```
   VARY SYSLOG,HARDCPY,OFF,UNCOND
   ```

   If SYSLOG has been turned off UNCOND, MVS does not save WQEs for logging after the LOGLIM has been reached. Some messages may be lost, but the system stays up.

3. SYSLOG becomes inactive.

   ```
   WRITELOG CLOSE
   ```

4. Make the necessary repairs to SYSLOG.
5. The SYSLOG becomes active with the following command, but it is not the hardcopy medium. WTLs are logged now, but not WTOs.

   ```
   WRITELOG START
   ```

6. SYSLOG becomes the hardcopy medium with the following command:

   ```
   VARY SYSLOG,HARDCPY
   ```

7. Set LOGLIM to a normal number.

   ```
   K M,LOGLIM=6000
   ```

3.2 Open data set limits

OS/390 Release 6 removed the 10,000 DD limit. Since then, many installations, mainly DB2 installations, have experienced a large increase in the number of concurrent allocations. This increase has caused a constraint on storage below the 16 MB line.

z/OS Version 1 Release 2 addresses this problem by moving the following control blocks above the 16 MB line to offer relief to the storage constraint:

- Data Set Allocation Block (DSAB)
- TCT I/O Measurement table (TCTIOT)
- Data Set Name Table (DSNT)

Due to this change in this release, the following types of software that use the DSAB chain to search for certain devices or data sets need to be evaluated for coding changes:

- Vendor products
- The following installation exits:
  - SVC99
  - SMS ACS exit
  - DEVSERV
  - OPEN, CLOSE, EOV
- Resource managers at task and address space termination
- Subsystems that support SUBSYS on a DD statement
- Subroutines called in a variety of environments
- Performance monitors
- Debugging tools

3.2.1 DSAB in storage

Figure 3-2 on page 41 illustrates a dual DSAB chain.

The QDB has pointers to the first and last DSAB on both chains.

The below-the-line-only chain does not include the above the line DSABs. The “all” chain includes both above and below the line DSABs.

When a DSAB resides above the line, the pointers to it are only on the “all” chain, and the below the line pointers are null.

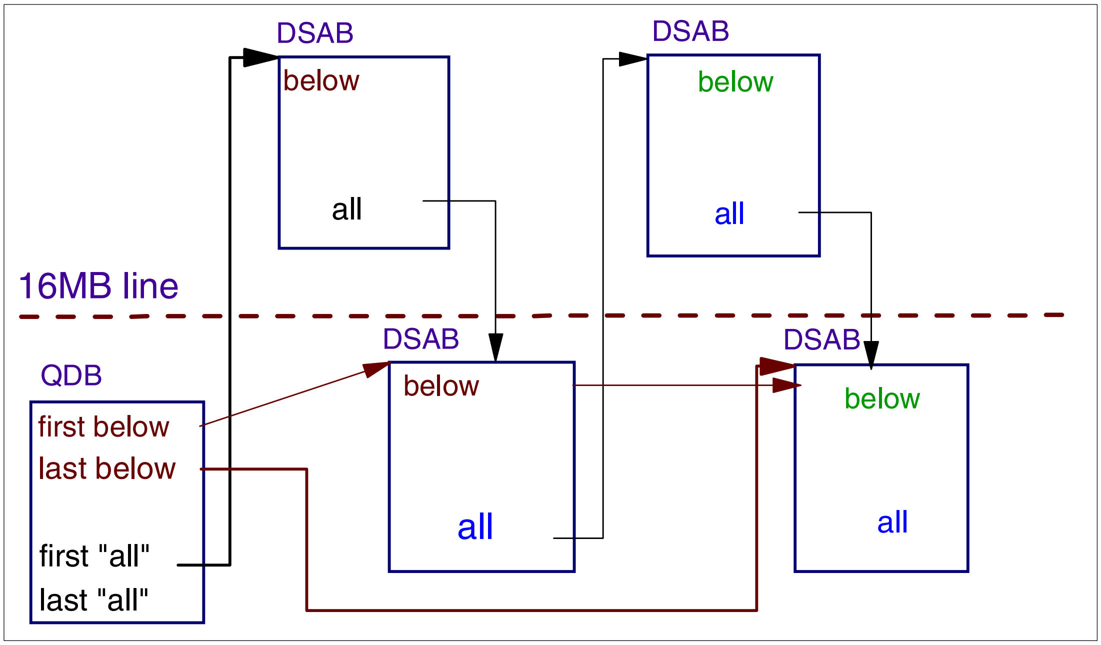


Figure 3-2 DSAB chaining in z/OS Version 1 Release 2

Note: All chains have a PREV and NEXT chain pointer.

3.3 Service aid enhancements

With z/OS Version 1 Release 2, the following enhancements are made to the MVS service aids. These enhancements are intended to offset the increasing complexity of installing and maintaining z/OS systems:

- IPCS enhancements:
  - When IPCS is invoked, it states which level of the operating system it is intended to support. This helps when debugging dumps or traces in a multi-system, multi-level environment.
  - IPCS allows users to access HFS paths.
  - IPCS uses data spaces to make more private area storage available for analysis and to accommodate larger reports.
  - The IPCS dialog provides a SORT primary command, which helps users manage multiple IPCS reports.
  - IPCS enhances the WHERE command to associate private area addresses with storage subpools.
  - IPCS eases problems when multiple dumps are required to adequately analyze a problem.
- Starting with z/OS V1R2, IBM supplies a large set of sample DUMP command parmlib members in SYS1.SAMPLIB. Each of the parmlib members can be used as supplied by IBM, or can be used as a base for further modification depending on installation-specific requirements, such as system names, address space names, and so on. To use these parmlib members, we recommend that you copy them to a data set in your parmlib concatenation. Care has been taken to ensure that system symbols are used where names can vary by installation.

- The SLIP command has additional parameters:
  - ACTION=STOPGTF
  - MSGID=

3.4 EOM resource manager changes

The End Of Memory (EOM) resource manager has experienced problems in the past that have usually been caused by serialization issues in the component that owns the problem EOM resource manager. These problems have caused termination of an address space to hang, possibly leading to inability to:

- Restart a terminating program
- Loss of function until the next IPL
- A loss of the system

These problems are addressed in this release by the CALLRTM program, which has also been changed to eliminate past problems.

Now, the EOM resource manager timeout function monitors possible problems every four minutes and examines the EOM resource managers to see if they have been dispatched since the timeout function last looked at them. If they are considered to be in a hang condition, further action is taken. The new function takes a synchronous SVCDUMP to capture the hang condition and issues an ABEND30D to drive the resource manager's recovery routines. The ABEND30D will not be seen in this SVCDUMP, as it is taken before the abend is issued.

3.5 GRS wildcard support

Global Resource Serialization (GRS) has the ability to modify the ENQ/DEQ requests through the specification on Resource Name Lists (RNLs). With z/OS Version 1 Release 2, GRS RNL wildcard support is introduced to allow specification of wildcard characters (*,?) within the resource names.

GRS RNL wildcard support comprises the set of functions that are required to enable an installation to more effectively use the GRS RNL functions to customize ENQ/DEQ processing in a sysplex environment. The GRS functions that are changed in this release are:

- RNL wildcard support

  New support is added to enable wildcard matching (*,?) for the QNAME and RNAME keywords.

- ENQ/DEQ installation exit ISGNQXIT

  A new exit, ISGNQXIT, is added to drive a set of installation exits prior to RNL processing. GRS provides a general ENQ/DEQ exit, which allows an installation to safely modify ENQ/DEQ processing.

- Miscellaneous changes
- Compatibility support

  All of the functional changes provided by this support are provided in PTFs to prior releases to allow downlevel systems to coexist in GRS complexes with wildcarded RNLs.

3.5.1 RNL wildcard support

Global Resource Serialization now provides the ability to modify ENQ/DEQ requests with the specification of Resource Name Lists that include wildcard characters. When specifying these lists, an installation can alter the scope (either SYSTEM or SYSTEMS) of a request, or alternatively, indicate that a request should not be associated with a hardware reserve.

With sysplex and the growth in the number of components that use GRS services, it has become necessary to make the RNL mechanism more flexible due to the growing number of resources for which prefix matching is insufficient. For example, if an installation has a parmlib that is accessible to every system (“SYS1.PARMLIB”) and a parmlib that is only accessible by the local system (“SYS1.&sysname.PARMLIB”), it would be difficult to specify these with the current RNL definitions. Every time a new system is added to the sysplex complex, the RNLs would have to be updated to add a new specific entry for the new system. Many new resources merely have altered values in the middle of the resource name.

RNL type specification

Currently, RNL processing allows for only an exact match or a prefix match, via the TYPE specification.

Table 3-1 RNL type specifications prior to this release

SPECIFIC	Indicates that the QNAME and RNAME defined in the RNLDEF entry must match exactly the resource name specified on the ENQ/DEQ macro.
GENERIC	Indicates that the resource name in the RNLDEF entry must be a prefix match of the resource name specified on the ENQ request.

Wildcard characters

This support adds the capability to enable matching on a set of characters within the QNAME or RNAME specification by including specification of the wildcard characters, shown in Table 3-2.

Table 3-2 Wildcard characters

*****	Allows matching for a substring of any characters for any lengths, including zero.
?	Allows matching for any single character.

New RNL type specification

To be able to use wildcard characters, GRS RNL wildcard introduces a new type parameter that can be specified in the TYPE keyword of the RNLDEF entry identifier, shown in Table 3-3, to differentiate the matching specification.

Table 3-3 Pattern definition

PATTERN	Indicates that the resource name in the RNL entry is a pattern which must fit the resource name specified on the ENQ request.

Note: PATTERN can be abbreviated as PATT.

Resource names with wildcards

With this new support for wildcards, the RNLDEF statement has been updated as shown in Figure 3-3 on page 44.

```
RNLDEF

RNL(INCL | EXCL | CON)

TYPE(SPECIFIC | GENERIC | PATTERN)

QNAME(qname)

RNAME(rname)
```

Figure 3-3 RNLDEF statement

Entries in an RNL definition can be specific or generic, or can be specified through the use of wildcard characters with TYPE(PATTERN). The types of resource definitions are as follows:

SPECIFIC A specific resource name entry matches a search argument only when they are exactly the same.

GENERIC A generic resource name entry is a portion of a resource name. A match occurs whenever the specified portion of the generic resource name entry matches the same portion of an input search argument.

PATTERN A pattern resource name entry, containing wildcard characters, extends the matching specification. The wildcard characters (*,?) can be used within both parts of the resource name.

> **Note:** After all the definitions are implemented and the system is running, all of the specific entries in a RNL are searched before scanning the generic and pattern RNL entries for a match. If no specific entry matches, the first generic or pattern entry that matches is used.

RNLDEF examples

Consider an installation that has a three system sysplex with the local and global resources described in Table 3-4.

Table 3-4 Global and local resource definitions

System Name	Local Resources	Global Resources
SC63	SYS1.SC63.LOGREC SYS1.SC63.MANX	SYS1.PROD.NUCLEUS SYS1.PROD.LINKLIB SYS1.PROD.LPALIB SYS1.SC63.LINKLIB SYS1.SC63.LPALIB SYS1.SC64.LINKLIB SYS1.SC64.LPALIB
SC64	SYS1.SC64.LOGREC SYS1.SC64.MANX	
SC65	SYS1.SC65.LOGREC SYS1.SC65.NUCLEUS SYS1.SC65.LINKLIB SYS1.SC65.LPALIB SYS1.SC65.MANX	

Without wildcard support

Without GRS RNL wildcard support, the installation would need to use the specification shown in Figure 3-4 on page 45. This specification is restricted by the need to have entries that distinguish each system name qualifier in the data set name. If the installation were to add a new system to the sysplex, say SC66, then the RNLs would have to be updated prior to IPLing the new system into the complex.

```
/* SYSTEM INCLUSION LIST */
RNLDEF RNL(INCL) TYPE(GENERIC)
QNAME(SYSDSN) RNAME(SYS1.PROD)
RNLDEF RNL(INCL) TYPE(GENERIC)
QNAME(SYSDSN) RNAME(SYS1.SC63)
RNLDEF RNL(INCL) TYPE(GENERIC)
QNAME(SYSDSN) RNAME(SYS1.SC64)

/* SYSTEM EXCLUSION LIST */
RNLDEF RNL(EXCL) TYPE(SPECIFIC)
QNAME(SYSDSN) RNAME(SYS1.SC63.LOGREC)
RNLDEF RNL(EXCL) TYPE(SPECIFIC)
QNAME(SYSDSN) RNAME(SYS1.SC64.LOGREC)
RNLDEF RNL(EXCL) TYPE(SPECIFIC)
QNAME(SYSDSN) RNAME(SYS1.SC63.MANX)
RNLDEF RNL(EXCL) TYPE(SPECIFIC)
QNAME(SYSDSN) RNAME(SYS1.SC64.MANX)
RNLDEF RNL(EXCL) TYPE(GENERIC)
QNAME(SYSDSN) RNAME(SYS1.TEST)
```

Figure 3-4 RNLs before GRS RNL wildcard support

With wildcard support

With GRS RNL wildcard support, the installation could define the RNLs as in Figure 3-5, allowing the new system to enter the sysplex without requiring a change to the RNL specification.

```
/* SYSTEM INCLUSION LIST */
RNLDEF RNL(INCL) TYPE(PATTERN)
QNAME(SYSDSN) RNAME(SYS1.*.*)

/* SYSTEM EXCLUSION LIST */
RNLDEF RNL(EXCL) TYPE(PATTERN)
QNAME(SYSDSN) RNAME(SYS1.*.LOGREC)
RNLDEF RNL(EXCL) TYPE(PATTERN)
QNAME(SYSDSN) RNAME(SYS1.*.MANX)
RNLDEF RNL(EXCL) TYPE(GENERIC)
QNAME(SYSDSN)  RNAME(SYS1.TEST)
```

Figure 3-5 RNLs with GRS wildcard support

3.5.2 ENQ/DEQ exit

Currently, GRS provides what is called a “RNL Exit,” which is called by the system to allow an installation to modify RNL processing. However, this exit is linked into the nucleus (IEANUC0x), forcing a re-IPL of the system to make any changes. This exit is also only capable of making minor changes to ENQ processing since it actually uses the caller’s parameter list as input. A poorly coded RNL exit can cause a system to enter into a disabled waitstate or resource integrity errors to occur.

With GRS RNL wildcard support, GRS provides a general ENQ/DEQ exit that enables the installation to safely modify ENQ/DEQ processing in the following ways:

- Change the resource name (QNAME and/or RNAME)
- Change the resource SCOPE
- Change the UCB address (for a RESERVE)

- Indicate to convert a RESERVE to an ENQ
- Indicate to convert an ENQ to a RESERVE (add a UCB specification)
- Indicate to bypass RNL processing

The ENQ/DEQ exit is only invoked for SCOPE=SYSTEM or SYSTEMS and not for STEP requests.

> **Note:** RNL(NO) requests will be passed to the exit, but a scope change requested by the exit will *not* be honored, nor will the request be passed to RNL processing.

The ENQ/DEQ exit is only invoked on the system where the ENQ or DEQ request is issued. The exit is not invoked on other systems in the GRS complex for global resource requests.

The ENQ/DEQ exit is installed via standard dynamic exits support.

Existence of old exit

This new support detects the existence of an installation installed version of the “old” RNL exit in SYS1.NUCLEUS.

In prior releases, the installation was able to install modified versions of the RNL search routines under the following entry points:

ISGGSIEX	System inclusion entry
ISGGSEEX	Systems exclusion entry
ISGGRCEX	Reserve conversion entry

During GRS initialization, if these entries are detected, an error message is issued indicating that they are not invoked, as follows:

```
ISG351I GLOBAL RESOURCE SERIALIZATION RNL EXIT exitname DETECTED.
         THIS EXIT IS NO LONGER SUPPORTED.
```

Explanation: During system initialization, Global Resource Serialization detected the presence of the specified RNL exit. This exit is no longer supported by Global Resource Serialization.

In the message text:

exitname - Specifies the name of the detected RNL exit, one of:

ISGGSIEX - RNL exit routine for SYSTEM inclusion RNL

ISGGSEEX - RNL exit routine for SYSTEMS exclusion RNL

ISGGRCEX - RNL exit routine for RESERVE conversion RNL

System Programmer response: Remove the specified exit from the system. If the exit provides a required function, implement the function via the ISGNQXIT dynamic exit point.

> **Note:** ISG351I is introduced to alert customers who are using the RNL exits that these exits are no longer supported. The support does not remove the exits because customers who have replaced the default exits with their own exits need to see this message. If you have not modified the default RNL exits, then either remove the exit or simply ignore the message.

3.5.3 Miscellaneous changes

The following changes have been made in support of the major external changes:

- The GRS monitor tool

  The monitor displays are updated to indicate if the request was altered by one of the ENQ/DEQ exit routines.

- Four new entries have been added to the list of authorized resource names that may only be used by authorized callers:

  – ADRDFRAG
  – ADRDSN
  – ARCENQG
  – BWODSN

3.5.4 Usability support

Installations exploiting z/OS Release 2 are not required to alter any of the current RNL specifications. However, with the introduction of the GRS RNL wildcard support, installations may wish to simplify their RNL lists by using the new pattern specification. There are two primary areas that can be simplified:

1. RESERVE processing

   In a Parallel Sysplex that has implemented GRS star complex for serialization, the recommendation is that all sysplex-wide RESERVEs be converted to GRS global ENQs, by using the Reserve conversion RNL. There are significant advantages to be gained in performing this:

   Performance Star mode global ENQs are faster than hardware reserves.

   Serviceability GRS global ENQs are tracked across the sysplex. With the DISPLAY, GRS,ANALYZE command, an installation can quickly determine the source of resource lockouts and take action when deadlocks occur (deadly embraces).

   Availability ENQs have a much finer granularity, locking a single data set, rather than RESERVE, which locks a complete DASD unit. This finer-grained serialization provides greater deadlock avoidance for the RESERVE. To implement conversion of all RESERVEs in a sysplex, code the following as the only RESERVE conversion entry in the GRSRNLxx member of parmlib:

   ```
   RNLDEF RNL(CON) TYPE(PATTERN) QNAME(*)
   ```

   Note: This is only recommended for installatons using GRS star.

2. Resource names that match in a pattern that cannot be easily specified with prior support capability.

   The recommendations for some data set naming conventions follow a pattern of “SYS*n.sysname.dataset.”*

3.5.5 Compatibility support

All of the functional changes can be rolled back to the prior releases to ensure compatibility. This support is available on OS/390 R8 through R10 and z/OS V1R1 via a PTF for APAR OW49779.

This support runs in any GRS mode as follows:

- NONE (ENQ/DEQ exit only)
- RING
- STAR

Coexistence support

Systems with wildcard support installed can tolerate being in a sysplex with systems that do not have the PTF for APAR OW49779 installed as follows:

- Mixed z/OS and OS/390 levels are acceptable.
- Do not define PATTERN entries in the RNLs if the PTF is not installed on all systems. This can have unpredictable results, such as resources not being correctly serialized or an 0A3 waitstate.

For more information on the new changes to GRS, see *z/OS MVS Planning: Global Resource Serialization*, SA22-7600.

3.6 DASD sharing between sysplexes

It is possible to support shared DASD with z/OS and OS/390 systems across GRS-plexes, as shown in Figure 3-6, by using the exit code provided in Appendix A, “Sample GRS exit ISGNQXIT” on page 297, and placing it in the new GRS ISGNQXIT exit.

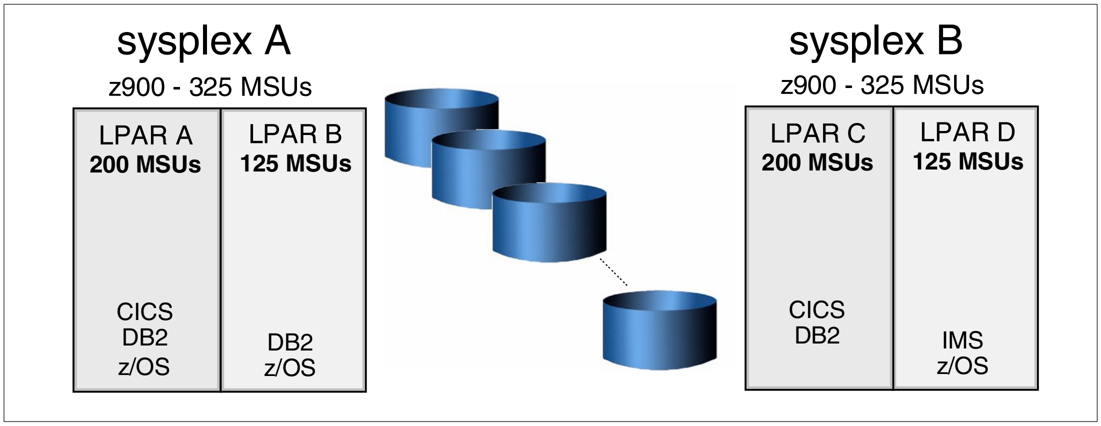


Figure 3-6 Shared DASD between sysplexes

This exit code allows all RESERVE requests for specified volumes in the RNL list to result in an HW RESERVE whenever the resource name is specified in the RNL list. Using the new ISGNQXIT exit and by adding an RNL definition to the conversion table with a QNAME not used by the system and RNAMEs that identify the volumes to share outside of GRS, the RESERVE requests for the volumes included in the RNL definitions will always result in an HW Reserve.

Restriction: The exit, however, does not propagate cross-sysplex ENQ requests with scope=SYSTEMS. For additional information, see “Exit restrictions” on page 50.

```
RNLDEF RNL(CON) TYPE(SPECIFIC|GENERIC)
QNAME(HWRESERV)
RNAME(VOLSER|volser-prefix)
```

Figure 3-7 Sample RNL list to define sysplex DASD sharing

The same RESERVE resource requests, which address other volumes, should have the possibility to be filtered through the conversion RNLs to have the HW reserve eliminated.

3.6.1 GRS ISGNQXIT exit

ISGNQXIT exit receives control for all RESERVE-ENQ-DEQ macros before the GRS RNLs processing logic. The exit traps the ENQ requests with a UCB pointer (scope SYSTEMS and HW RESERVE), and bypasses all the other ENQ-DEQ requests.

The RNL exclusion list is scanned first to see if a specific or generic RNL entry matches the request's major-minor name; pattern type entries are bypassed. If a match is found, control is returned to GRS with no action. The ENQ or DEQ request is then filtered by GRS through the RNL exclusion table, and, because it matches, the scope is changed to SYSTEM with the HW RESERVE issued.

If the RNL exclusion table scan does not match major-minor name, then the exit locates the 'VOLSER' inside the parameter list and checks if the resource, HWRESERV, is present in the RNL conversion table; only specific and generic type entries are checked, pattern type entries are bypassed.

If a match is found, the bypass RNL processing bit is set in the parameter list and control is returned to the system. The GRS RNL's processing is not done, the request scope remains SYSTEMS and the HW RESERVE is issued. Some overhead is paid because a double serialization is used instead of one.

The reason the exit should do the RNL exclusion table scan is because it is legal, since OS/360, to issue a RESERVE macro (ENQ scope SYSTEMS with UCB pointer that implies a HW Reserve), and remove the RESERVE request with a DEQ for the same major-minor name, scope SYSTEMS, with or without UCB pointer.

A RESERVE request trapped by the exit has scope SYSTEMS plus HW Reserve, and may be ended with a DEQ scope SYSTEMS without a UCB pointer. In this scenario, without the RNL exclusion table scan, a DEQ with a major-minor name present in the RNL exclusion table would have the scope changed to SYSTEM by GRS RNL processing, and the DEQ would not remove the HW RESERVE because it is associated to a request with scope SYSTEMS. This could cause the DEQ to abend with a system code 130.

For example, RESERVEs for major name SYSIGGV2 are removed with DEQs with scope=SYSTEMS without a UCB pointer. Therefore, the RNL exclusion table scan prevents the exit from trapping RESERVE requests that will be excluded by GRS RNL processing, and therefore becoming scope=SYSTEM with a HW RESERVE.

Because the RNL conversion table is used, any change to the table can be activated through the following z/OS operator command:

```
SET GRSRNL=xx
```

It is recommended that the VOLUME(S) behind the QNAME(HWRESERV) should not be dynamically changed unless the devices are off-line or not allocated.

The qname 'HWRESERV', shown in Figure 3-8, is hard-coded and is defines at label HRDWNAME in the ISGNQXIT example.

Figure 3-8 shows an example of a RNL conversion table to cross Sysplex HW RESERVE volume XA9RES and volumes with prefixes of CIX.

```
RNLDEF RNL(CON) TYPE(GENERIC)
QNAME(SYSVTOC)

RNLDEF RNL(CON) TYPE(GENERIC)
QNAME(SPFEDIT)
RNLDEF RNL(CON) TYPE(GENERIC)
QNAME(SYSIGGV2)

RNLDEF RNL(CON) TYPE(PATTERN)      /*this entry is bypassed*/
QNAME(ARC*)                        /*by the exit */

RNLDEF RNL(CON) TYPE(SPECIFIC)
QNAME(HWRESERV)                     /*SPECIAL NAME*/
RNAME(XA9RES)                       /*ALWAYS RESERVE XA9RES*/

QNAME(HWRESERV)                     /*SPECIAL NAME*/
RNAME(CIX)                          /*ALWAYS RESERVE VOLUMES*/
                                    /*BEGINNING WITH CIX   */
```

Figure 3-8 RESERVE conversion resource name list example

Exit restrictions

The exit has some restrictions that should be understood and evaluated very carefully, as follows:

- The exit does not propagate cross-sysplex global ENQs (scope=SYSTEMS), it only guarantees that the HW RESERVEs are issued for volumes behind qname HWRESERV in the RNL conversion table for all RESERVE requests. Applications that logically serialize a DASD resource (PDS member or data set) with a global ENQ (scope=SYSTEMS) cannot depend on the exit to serialize resources on DASD shared between sysplexes.
- Type PATTERN is not supported for HWRESERV RNL entries.
- If EXCLUSION RNL entries are used to have RESERVE requests handled with SCOPE=SYSTEM and HW RESERVE for volumes shared between GRS-plexes, type PATTERN entries are not supported.
- It is possible to dynamically add and remove Volumes behind QNAME(HWRESERV) with the SET GRSRNL=xx z/OS command, but the system does not check if the Volumes have outstanding RESERVE requests before activating the new RNLs. It is recommended that the volume(s) specified behind the QNAME(HWRESERV) should not be dynamically changed unless the device is off-line or not allocated.

Exit compatibility

To support sysplex DASD sharing in configurations where the sysplexes may have GRS with and without wildcard support, the ISGNQXIT dynamic exit is compatible with the ISGGREX1 ITSO exit provided for MVS and OS/390 systems without GRS wildcard support. Both exits have the same scope and functionality. The ISGNQXIT exit is a replacement for the ISGGREX1 exit beginning with z/OS Version 1 Release 2.

3.6.2 Exit installation and activation

The exit can be activated by one of the following methods:

- With a program using CSVDYNEX macro.
- With the SETPROG EXIT z/OS operator command:

```
SETPROG EXIT,ADD,EX=ISGNQXIT,MOD=ISGNQXIT,DSN=SYS1.USER.LINKLIB
```

- Using an EXIT STATEMENT of the PROGXX parmlib member:

```
EXIT ADD
EXITNAME(ISGNQXIT)
MODNAME(ISGNQXIT)
STATE=ACTIVE
DSNAME(SYS1.USER.LINKLIB)
```

The recommended procedure to activate the exit after an IPL is the following:

1. Activate the exit in all systems sharing DASD across the sysplexes.
2. Update the RNL conversion table in the GRSRNLxx parmlib member.
3. Activate the RNLs in all systems sharing DASD across the sysplexes.

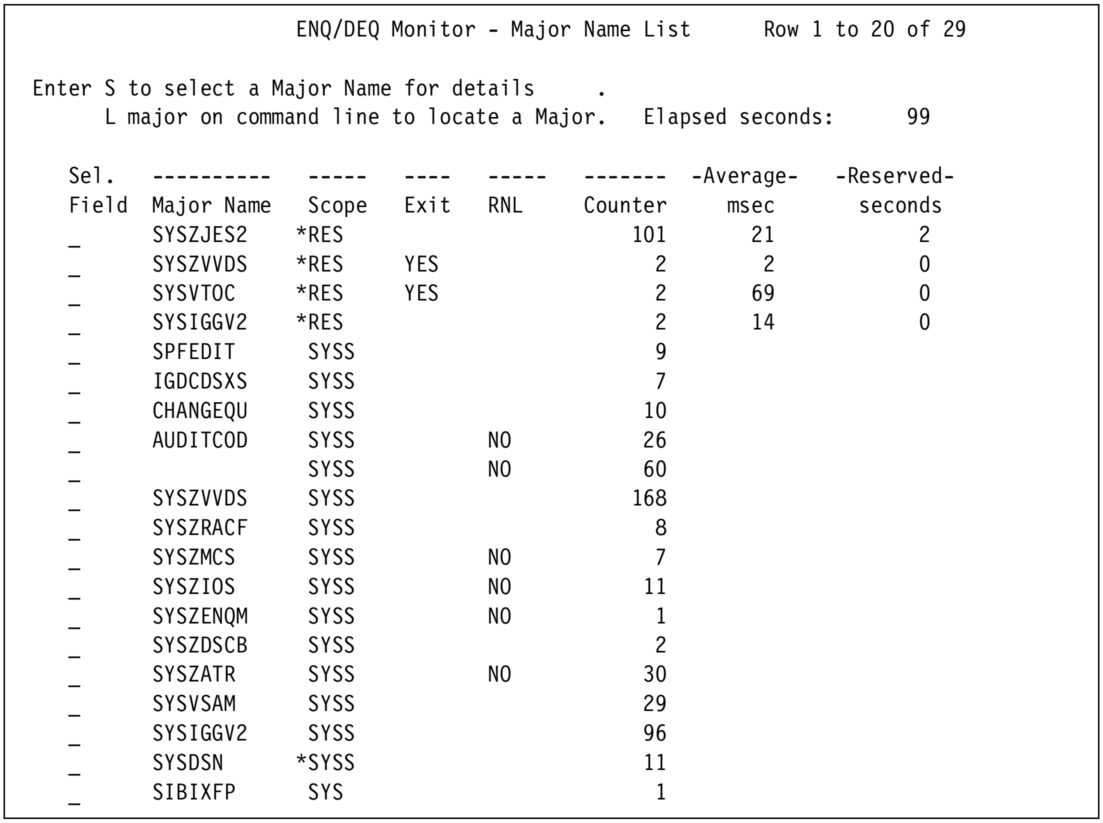
For additional information, you can refer to *z/OS MVS Installation Exits*, SA22-7593.

EXIT process verification

To verify if the exit is behaving as expected, you can use the ENQ/RESERVE/DEQ Monitor. Monitor selection 1, MAJOR Names Display, has been extended with a new column to indicate if the request has been modified by exit ISGNQXIT.

Monitor selection 1

Figure 3-9 shows a MAJOR Names display example.



```
                       ENQ/DEQ Monitor - Major Name List      Row 1 to 20 of 29

 Enter S to select a Major Name for details     .
       L major on command line to locate a Major.   Elapsed seconds:      99

    Sel.   ----------   -----   ----   -----   -------  -Average-   -Reserved-
    Field  Major Name   Scope   Exit   RNL     Counter     msec       seconds
    _      SYSZJES2    *RES                        101        21           2
    _      SYSZVVDS    *RES     YES                  2         2           0
    _      SYSVTOC     *RES     YES                  2        69           0
    _      SYSIGGV2    *RES                          2        14           0
    _      SPFEDIT      SYSS                         9
    _      IGDCDSXS     SYSS                         7
    _      CHANGEQU     SYSS                        10
    _      AUDITCOD     SYSS           NO           26
    _                   SYSS           NO           60
    _      SYSZVVDS     SYSS                       168
    _      SYSZRACF     SYSS                         8
    _      SYSZMCS      SYSS           NO            7
    _      SYSZIOS      SYSS           NO           11
    _      SYSZENQM     SYSS           NO            1
    _      SYSZDSCB     SYSS                         2
    _      SYSZATR      SYSS           NO           30
    _      SYSVSAM      SYSS                        29
    _      SYSIGGV2     SYSS                        96
    _      SYSDSN      *SYSS                        11
    _      SIBIXFP      SYS                          1
```

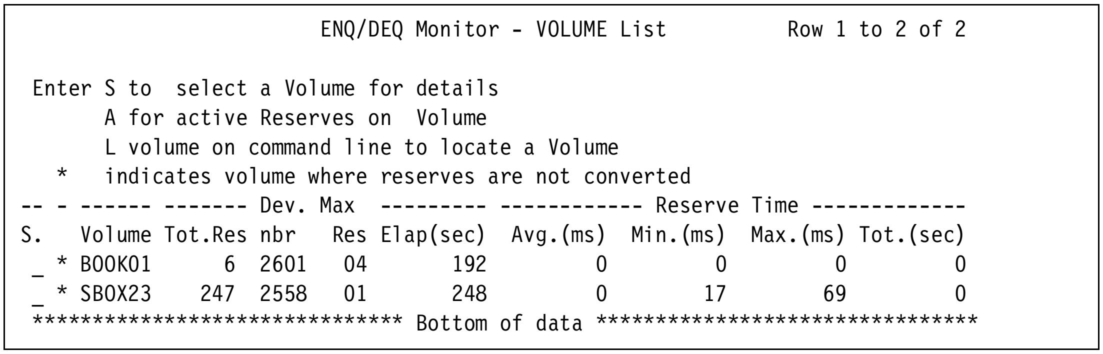
Figure 3-9 ENQ/DEQ Monitor selection 1

Figure 3-9 shows that RESERVE requests for Major names SYSZVVDS and SYSVTOC have been processed by exit ISGNQXIT and that the HW Reserves have been issued.

Monitor selection 3

Using Monitor selection 3 and the VOLUME list display, you can see the volumes where HW Reserves have been issued. Figure 3-10 shows two volumes:

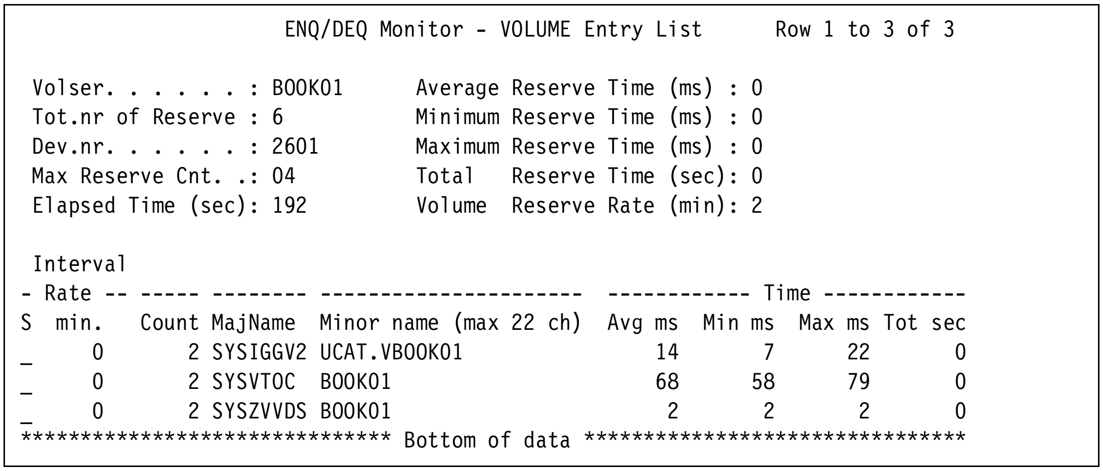
- BOOK01 shared across two sysplexes and therefore with a HWRESERV entry in the RNL conversion table
- SBOX23 with JES2 checkpoint that has its RESERVE major name in the RNL Exclusion table



```
                         ENQ/DEQ Monitor - VOLUME List          Row 1 to 2 of 2

 Enter S to  select a Volume for details
       A for active Reserves on  Volume
       L volume on command line to locate a Volume
   *   indicates volume where reserves are not converted
-- - ------ ------- Dev. Max  --------- ------------ Reserve Time -------------
S.   Volume Tot.Res nbr   Res Elap(sec)  Avg.(ms)  Min.(ms)  Max.(ms) Tot.(sec)
 _ * BOOK01       6 2601   04       192         0         0         0         0
 _ * SBOX23     247 2558   01       248         0        17        69         0
 ******************************* Bottom of data ********************************
```

Figure 3-10 Volume List display with monitor selection 3

Selecting the entry for volume BOOK01, the major-minor name combinations of the RESERVE macros are displayed, as shown in Figure 3-11.



```
                      ENQ/DEQ Monitor - VOLUME Entry List      Row 1 to 3 of 3

 Volser. . . . . . : BOOK01      Average Reserve Time (ms) : 0
 Tot.nr of Reserve : 6           Minimum Reserve Time (ms) : 0
 Dev.nr. . . . . . : 2601        Maximum Reserve Time (ms) : 0
 Max Reserve Cnt. .: 04          Total   Reserve Time (sec): 0
 Elapsed Time (sec): 192         Volume  Reserve Rate (min): 2

 Interval
- Rate -- ----- -------- ----------------------  ------------ Time ------------
S  min.   Count MajName  Minor name (max 22 ch)  Avg ms  Min ms  Max ms Tot sec
_     0       2 SYSIGGV2 UCAT.VBOOK01                14       7      22       0
_     0       2 SYSVTOC  BOOK01                      68      58      79       0
_     0       2 SYSZVVDS BOOK01                       2       2       2       0
******************************* Bottom of data ********************************
```

Figure 3-11   BOOK01 detail display

For additional information about the GRS monitor, refer to *z/OS MVS Planning: Global Resource Serialization*, SA22-7600.

4

JES2 enhancements

This chapter describes the changes made to JES2 in z/OS Version 1 Release 2.

The major themes for this release are increased capacity and continuous operations. This chapter covers the support for all function changes introduced by this release of z/OS 1.2, as follows:

- Greater than 64K jobs.
- A new $ACTIVATE level z2.
- The number of JOBs, JOEs (output groups), BERTs, and track groups were all increased.
- Better management of JES log data sets for long running jobs.
- Dynamic PROCLIB and a new INCLUDE initialization statement, reducing the need to update the JES2 startup PROC.
- Spool now supports data sets up to 64K tracks placed anywhere on a volume.
- Installation consideration.

The support for greater than 64K jobs and the potential impact on vendor and customer applications is of particular importance.

4.1 Greater than 64K jobs

Many large installations are hitting JES2 constraints that limit the number of jobs they can have in their JES2 MAS. The major limit is the number of JQEs that JES2 can support. However, as the number of JQEs is increased, so too must the number of JOEs and BERTs be increased. The difficulty is that the maximum for each of these values (except for BERTs) are already at their architectural limits. This includes the size of the checkpoint data set. To increase the limits, a number of data structures had to be reorganized. As installations combine more systems into a single sysplex and into a single MAS, the number of jobs that are concurrently resident on the JES2 spool increases.

Before this Release of JES2, JES2 can support up to the following limits:

- JQEs up to 65,534 jobs
- JOEs up to 161,314 output elements

In many large installations, these old limits are no longer enough. The number of JQEs and JOEs need to be increased to support the increased number of jobs that can reside in a JES2 MAS. The new limits that have increased on the JES2 initialization statements are as follows:

JOBDEF	JOBNUM limit is 200,000
JOBDEF	RANGE limit is 999,999
OUTDEF	JOENUM limit is 500,000
CKPTSPACE	BERTNUM limit is 500,000
SPOOLDEF	TGSPACE=MAX limit is 16,580,355

These new limits that are increased to new maximum values are not the new architecture limits, but rather values that JES2 can comfortably accommodate.

The JOBDEF RANGE supports more values than the current JOBDEF JOBNUM limit. This was done so that customers that run a large number of jobs do not wrap the job numbers as frequently.

The values for JOBNUM, JOENUM, and BERTNUM can now be decreased via operator commands. This is true in both R4 and z2 mode, but can only be done from a z2 level of JES2.

4.1.1 $ACTIVATE command

The **`$ACTIVATE`** command expands the JES2 checkpoint to enable functions introduced in Release 4, or now in z/OS Version 1 Release 2. Once the checkpoint has been expanded and the new functions enabled, it is possible to undo the effects of the **`$ACTIVATE`** command via a JES2 cold start at a prior release level, or by using the UNACTIVATE start option during a JES2 all-member restart.

A new level of **`$ACTIVATE`** is available to support the increase of job numbers to a maximum of 999,999. This new **`$ACTIVATE`** level is called Release z2 mode and when activated, it allows you to exploit new JES2 functions using job numbers greater than 65534. All members of the MAS must be at this Release z2 mode level or higher when the **`$ACTIVATE`** is performed.

Display current mode

The **$ACTIVATE** command has a required subparameter in which you specify the level you intend to activate.This is to prevent the chance of a mis-typed **$DACTIVATE** (drop of the '**D**') accidentally activating a new level. This request was the result of installations wanting to do a **$DACTIVATE**, shown in Figure 4-1, to determine the current checkpoint level and forgetting the '**D**'.

```
$DACTIVATE
$HASP895 $DACTIVATE 513
$HASP895 JES2 CHECKPOINT LEVEL IS NOW OS/390 RELEASE 4
$HASP895 A TOTAL OF 1087 4K RECORDS ARE REQUIRED FOR $ACTIVATE.
$HASP895 ALL INUSE=YES DATA SETS ARE AVAILABLE AND LARGE ENOUGH.
$HASP895 $ACTIVATE WILL SUCCEED IF ISSUED FROM THIS MEMBER
```

Figure 4-1 $DACTIVATE command to display the current checkpoint mode

Note: The response to the $DACTIVATE command specifies how much checkpoint space to allocate, by the number of 4K pages needed, when you issue the $ACTIVATE command.

$ACTIVATE subparameter

Now, at JES2 1.2 level, you must specify a subparameter on the command or you receive an error message and the activate is not accomplished, as shown in Figure 4-2.

```
$ACTIVATE
$HASP003 RC=(12),IVATE  - MISSING REQUIRED OPERAND LEVEL
$HASP895 JES2 CHECKPOINT LEVEL IS NOW OS/390 RELEASE 4
```

Figure 4-2 $ACTIVATE command without the subparameter

For example, you must specify the level subparameter to convert to the new checkpoint mode, as shown in Figure 4-3, to activate your MAS in Release z2 mode.

```
$ACTIVATE,LEVEL=Z2
$HASP895 z/OS 1.2 LEVEL IS NOW ACTIVE
$HASP895 JES2 CHECKPOINT LEVEL IS NOW z/OS 1.2
```

Figure 4-3 $ACTIVATE command to convert to z2 mode

Note: Once activated, no level of JES2 prior to JES2 z/OS V1R2 is allowed to join the MAS, nor is it allowed to start using the JES2 z/OS V1R2 checkpoint. At cold start, JES2 starts in Release z2 mode.

In "Release 4 mode," JES2 operates as in JES2 z/OS V1R1 and all previous releases of OS/390 back to Release 4.

Return to R4 mode

The **$ACTIVATE** command can also be used to revert back to R4 mode from z/OS1.2 mode. The command to convert back to release 4 mode is **$ACTIVATE,LEVEL=R4**, as shown in Figure 4-4 on page 58.

```
$ACTIVATE,LEVEL=R4
$HASP895 OS/390 RELEASE 4 LEVEL IS NOW ACTIVE
$HASP895 JES2 CHECKPOINT LEVEL IS NOW OS/390 RELEASE 4
```

Figure 4-4 $ACTIVATE command to convert back to R4 mode

Prior to performing the unactivation, a number of checks are made to ensure that no features that are only supported in z/OS1.2 mode are active. If any of the z/OS 1.2 functions are active, the **$ACTIVATE,LEVEL=R4** fails with a HASP445 message.

4.1.2 $ACTIVATE considerations

The recommendation is to not increase the new limits until the z2 mode is stable in the installation. If a problem is found after the limits have been increased, reduce the limits to a lower level or an R4 mode level rather than unactivating to the R4 mode.

Return to R4 mode

As in earlier releases, you can control what mode JES2 runs in using an option specified on the PARM= keyword when you start JES2. On a cold start, using the UNACT option on the PARM= causes JES2 to cold start in Release 4 mode.

```
PARM=(COLD,UNACT)
```

If UNACT is not specified, JES2 will cold start in z2 mode.

```
PARM=(WARM)
```

Using UNACT on an all member warm start or on an “all member hot” start will cause JES2 to switch to release 4 mode. There is no option to switch to z2 mode on a warm start.

```
PARM=(WARM,UNACT)
```

Note: The **$ACTIVATE** command is the preferred method of switching modes.

4.1.3 Maximum number of jobs

Once you are running in z2 mode, the job number range can be increased or decreased using the **$T** command. You can specify the maximum number (10-200000) of jobs that can be in the JES2 job queue at any given time. This value includes all TSU and STC jobs, as well as batch jobs.

```
$T JOBDEF,JOBNUM=200000
```

JES2 validates that the checkpoint is large enough to hold any increase in the size of the JQEs when you increase the JOBNUM value, as shown in Figure 4-5.

```
$T JOBDEF,JOBNUM=200000
$HASP296 MEMBER SC59 -- CKPT1 SYS1.HASPCKPT ON OP1TS2 - SPACE 592
         INSUFFICIENT -- 944 TRACKS NEEDED
$HASP003 RC=(74), 593
$HASP003 RC=(74),JOBDEF  - CURRENT CHECKPOINT DATA SETS ARE TOO
$HASP003         SMALL
```

Figure 4-5 Changing the JOBNUM to set maximum number of jobs

After the checkpoint data set is made larger and the command is entered again, the messages shown in Figure 4-6 on page 59 are received.

```
$T JOBDEF,JOBNUM=200000
$HASP835 JOBDEF
$HASP835 JOBDEF  ACCTFLD=REQUIRED,CNVT_ENQ=FAIL,JCLERR=YES,
$HASP835         JNUMBASE=48,JNUMFREE=32726,JNUMWARN=80,
$HASP835         JOBFREE=199959,JOBNUM=200000,JOBWARN=80,
$HASP835         PRTYHIGH=10,PRTYJECL=YES,PRTYJOB=YES,PRTYLOW=5,
$HASP835         PRTYRATE=144,RANGE=(1,32767),RASSIGN=YES,
$HASP835         DUPL_JOB=DELAY
```

Figure 4-6 Setting the JOBNUM value to 200000

4.1.4 Job number range

The range of job numbers has now changed to (nnnnnn-mmmmmm), which JES2 can use to assign JOBIDs to jobs when you specify a range, such as 1 to 999999:

1 The lowest number that is assigned as a JES2 job identifier for jobs

999999 The highest number that can be assigned as a JES2 job identifier

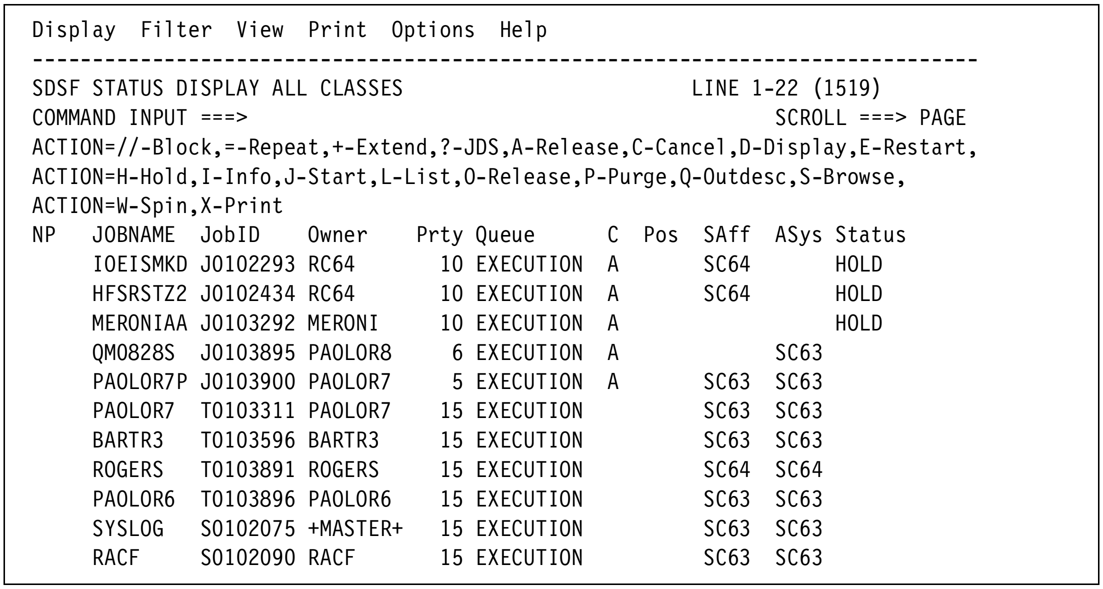
Note: The value specified for the highest number (mmmmmm) must be equal to the lowest number (nnnnnn) or at least 10 greater than the value used for the lowest number (nnnnnn).

Depending on when you change the range of job numbers, there may be existing jobs in the system.

JOBIDs are changed for the new limits when the JOBID is above 99,999, as follows:

- STC12345 now becomes S0123456
- JOB12345 now becomes J0123456
- TSU12345 now becomes T0123456

Examples of the new JOBIDs are shown in the SDSF status display in Figure 4-7.



```
 Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 SDSF STATUS DISPLAY ALL CLASSES                       LINE 1-22 (1519)
 COMMAND INPUT ===>                                           SCROLL ===> PAGE
 ACTION=//-Block,=-Repeat,+-Extend,?-JDS,A-Release,C-Cancel,D-Display,E-Restart,
 ACTION=H-Hold,I-Info,J-Start,L-List,O-Release,P-Purge,Q-Outdesc,S-Browse,
 ACTION=W-Spin,X-Print
 NP   JOBNAME  JobID    Owner    Prty Queue      C  Pos  SAff  ASys Status
      IOEISMKD J0102293 RC64       10 EXECUTION  A       SC64       HOLD
      HFSRSTZ2 J0102434 RC64       10 EXECUTION  A       SC64       HOLD
      MERONIAA J0103292 MERONI     10 EXECUTION  A                  HOLD
      QM0828S  J0103895 PAOLOR8     6 EXECUTION  A             SC63
      PAOLOR7P J0103900 PAOLOR7     5 EXECUTION  A       SC63  SC63
      PAOLOR7  T0103311 PAOLOR7    15 EXECUTION          SC63  SC63
      BARTR3   T0103596 BARTR3     15 EXECUTION          SC63  SC63
      ROGERS   T0103891 ROGERS     15 EXECUTION          SC64  SC64
      PAOLOR6  T0103896 PAOLOR6    15 EXECUTION          SC63  SC63
      SYSLOG   S0102075 +MASTER+   15 EXECUTION          SC63  SC63
      RACF     S0102090 RACF       15 EXECUTION          SC63  SC63
```

Figure 4-7 SDSF status display showing new JOBIDs

Jobs from NJE

Jobs that enter a JES2 V1.2 system can use the new JOBID format if the original job number is available and greater than 99999. However, you must specify in the JES2 initialization statements the following:

```
JOBDEF REASSIGN=YES
```

REASSIGN=YES specifies whether job numbers outside of the RANGE definition can be assigned to jobs received via NJE or spool reload. Jobs with original job numbers outside the JOBDEF RANGE will retain their original job number if possible. If you specify REASSIGN=NO, jobs with original job numbers outside the JOBDEF RANGE will always be assigned a new job number within the range.

4.2 Long running jobs support

Long running jobs (for example: VTAM, HSM, CICS, IMS, and NetView) use spool space for their JOBLOG and system messages data sets. The problem is, the longer these applications run, the larger these data sets become, and the more spool space they consume. Eventually, the space used by these system data sets can interfere with the normal operations of the system. The only solution is to either recycle the offending applications, or IPL the system. Even sending the output to a SYSOUT class defined as DUMMY does not prevent the writing of these data sets.

To solve this, JCL and JES2 externals are augmented to support the capability to spin off the JOBLOG and system messages data sets in the cases where this data is needed, and to totally suppress the writing of these data sets if they are not required. This eliminates the need to IPL to relieve a spool shortage due to long running jobs. This new function can be applied on a individual job basis, or by default, to all jobs in a job class.

The JESLOG (JES2 JESMSGLG and JESYSMSG) data sets for long running jobs use spool space which cannot be released until the job ends execution. Ending the execution of these types of jobs usually implies IPLing to free the spool space.

When you have jobs that run for extended periods of time, the JESLOG data needs to be written periodically as the job runs. JES2 in this release provides for the “spin off” or “suppression” of JESLOG data (specifically the data sets, JESMSGLG and JESYSMSG). This periodic ability to spin off JESLOG data improves the use of JES2 resources.

This release of JES2 allows the following options to control the JOBLOG data:

- JESLOG can be suppressed.
- JESLOG can be spun automatically based on line count or time interval or time of day.
- JESLOG can be spin eligible with support for an operator command that can be issued at any point in the execution of the job.

The ability to control what jobs or applications exploit this support is under installation control. No application changes are needed. Both JESLOG data sets must be processed in the same manner. Once an installation has decided to handle the JESLOG data sets differently, it must decide whether to suppress them or spin them. If JESLOG is to be spun, the mechanism to determine when to spin can be either a specific time each day, a time interval (such as once an hour) or based on size. It is possible to make the JESLOG data sets spinnable without specifying when to spin them. In this case, the data sets can be spun using an operator command.

This new function applies to all JES2 job types, batch jobs, TSO user jobs, and started tasks. It is expected that the most likely users are long running started tasks such as VTAM, HSM, NetView, and TCP/IP.

4.2.1 Execution phase processing

The JESLOG data set support applies only to the messages that are placed into the JESMSGLG and JESYSMSG data sets while the job is in execution.

Messages placed into this data set prior to execution (input processing, conversion processing, and via $LOGMSG) and after execution (output processing and job statistics) are placed into the traditional non-spin data sets and printed with the normal output of the job.

Even if no data sets are spun, the output of the job is affected when JESLOG=SPIN is specified. Messages that are issued during a jobs execution (to JOBLOG or system messages) appear separate from messages placed in the traditional JESLOG data sets.

SPIN messages

At the end of each spin segment, a HASP138 message, shown in Figure 4-8, indicates why that segment was spun (operator command, time interval, or line count). For each new SPIN data set for the job log, a new header line and date message is included. Each spin data set has a segment ID assigned (which can be printed on the separator page). Internally, the statically allocated data set is linked to the spin data sets, which are linked together. If the spool data set browse is used to browse one of the logical JESLOG data sets, then all the messages from the data set are displayed in chronological order. This feature can only be used if none of the spin data sets have been purged. If any data sets are purged, then the display ends when the current data set attempts to link to the purged data set.

```
$HASP138 JESLOG SPIN REQUESTED BY OPERATOR
$HASP138 JESLOG SPIN HAS OCCURRED BECAUSE OF TIME INTERVAL OR TIME OF DAY
$HASP138 JESLOG SPIN HAS OCCURRED BECAUSE OF LINE COUNT
```

Figure 4-8 Possible messages for JESLOG SPIN output

JESLOG spool space

The two data sets, JESMSGLG and JESYSMSG, are both spun when either data set satisfies the criterion or when an operator command is entered. The JESLOG data sets use space on the spool volumes even if a dummy MSGCLASS or NOLOG is specified on the /*JOBPARM JECL statement.

4.2.2 New JESLOG keyword

A new JESLOG= keyword on the JOBCLASS(v) initialization statement, along with the JESLOG= keyword on the job card, provides for the specification to spin off the JESLOG data sets.

> **Note:** The specification on the job card overrides what is specified on the JOBCLASS(v) initialization statement.

JOBCLASS initialization statement

The JOBCLASS initialization statement has a new keyword, JESLOG=, that specifies the following:

JESLOG=SPIN JESLOG is spin eligible.

JESLOG=SUPPRESS	JESLOG is suppressed.
JESLOG=NOSPIN	JESLOG is not spun (same as R10 and earlier).
JESLOG=(SPIN,n)	JESLOG automatically spun after n lines in one of the data sets. **n** is 500-999 or 1K to 999K or 1M to 999M.
JESLOG=(SPIN,time)	JESLOG automatically spun at time interval or time of day. **time** is hh:mm for time of day hh = 0 to 23 mm is 00 to 59. **time** is +hh:mm for time interval hh = 0 to 99 mm is 00 to 59. The minimum interval is +0:10.

JCL JOB statement

The JESLOG= keyword is new on the job statement. It has the same syntax as described for the JOBCLASS statement, except if time is specified. If time is specified, enclose time of day or elapsed specification in apostrophes (single quotes also accepted), as shown in the following:

```
//JOBNAME JOB,.....,JESLOG=(SPIN,'+5:13')
```

4.2.3 Processing spin output

The spool Data Set Browse interface is updated to allow the fetching of the logical JESMSGLG and the logical JESYSMSG data sets.

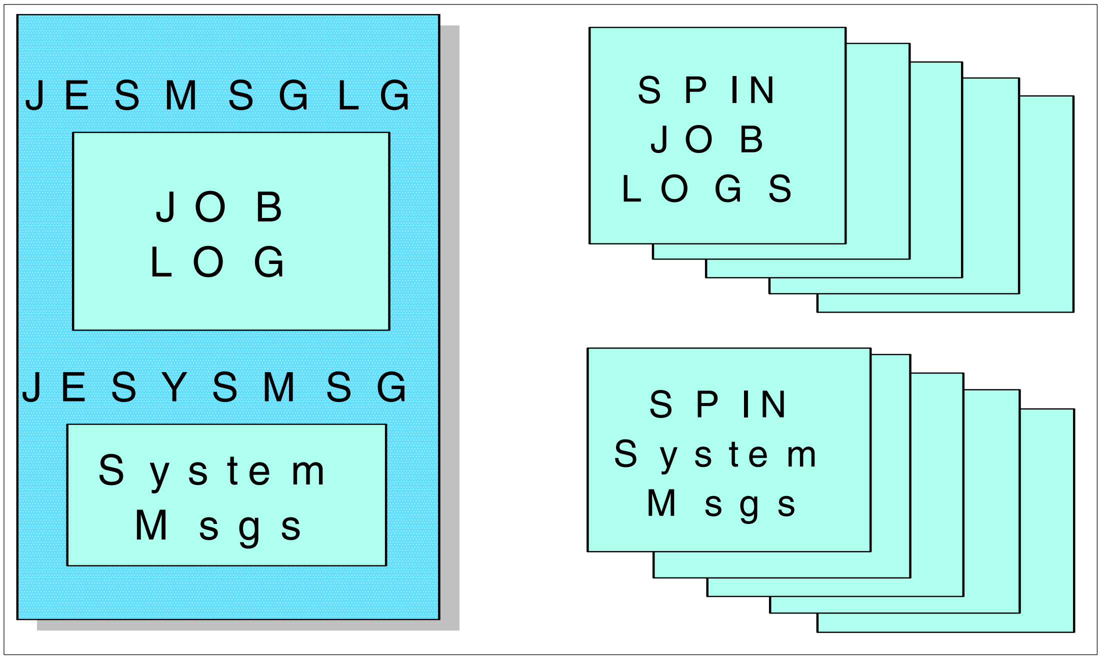


Figure 4-9 SPIN data sets for JESMSGLG and JESYSMSG

Operator commands

The **$TJ**, **$TT**, **$TS**, and **$TJOBQ** commands were updated to support a SPIN operand. These commands cause the target jobs to spin their JESLOG data sets immediately. This restarts any time interval or line interval specified on JESLOG=. Only jobs, STCs, or TSUs that have been specified as spinnable can be spun using the **$TJ(n),SPIN** commands. To determine if a job can be spun, the **$DJ(n),LONG** command now displays the SPINNABLE status of the job.

The operator can enter a command to change or to set the specifications of the:

- JOBCLASS initialization statement

  The **$T JOBCLASS(v)** command is available to alter your setting on the JOBCLASS(v) initialization statement.

  – **$T JOBCLASS(K),JESLOG=SPIN**

```
$TJOBCLASS(K),JESLOG=(SPIN)
$HASP837 JOBCLASS(A)
$HASP837 JOBCLASS(A)        ACCT=NO,PGMRNAME=NO,
$HASP837                    TIME=(000450,00),
$HASP837                    REGION=0002M,COMMAND=
$HASP837                    DISPLAY,BLP=YES,AUTH=(ALL),
$HASP837                    MSGLEVEL=(1,1),COPY=NO,
$HASP837                    HOLD=NO,IEFUJP=YES,IEFUSO=
$HASP837                    YES,JOURNAL=YES,LOG=YES,
$HASP837                    MODE=JES,OUTDISP=(,),
$HASP837                    OUTPUT=YES,PERFORM=000,
$HASP837                    PROCLIB=00,QHELD=NO,
$HASP837                    RESTART=YES,SCAN=NO,
$HASP837                    SCHENV=,SWA=ABOVE,TYPE26=
$HASP837                    YES,TYPE6=YES,XBM=,
$HASP837                    XEQCOUNT=(MAXIMUM=*,
$HASP837                    CURRENT=1),JESLOG=(SPIN)
```

Figure 4-10 Operator command to change the JESLOG option for job class k

- JESLOG options for a specific job

  – **$T J100,SPIN**

  – **$T S23165,SPIN**

  The **$T job SPIN=** command is provided so you can change the spin option or suppress the indication for a job. Figure 4-11 shows the STC job being changed to spin followed by the display long command, shown in Figure 4-12 on page 64, showing the STC as being changed to spinnable.

```
$TS(23165),SPIN
$HASP890 JOB(SDSF64)
$HASP890 JOB(SDSF64)    STATUS=(EXECUTING/SC64),
$HASP890                CLASS=STC,PRIORITY=15,
$HASP890                SYSAFF=(SC64),HOLD=(NONE)
```

Figure 4-11 Operator command to change a STC job's JESLOG option

After the command is entered, you can display the STC job as follows:

```
$DS(23165),LONG
$HASP890 JOB(SDSF64)
$HASP890 JOB(SDSF64)    STATUS=(EXECUTING/SC64),
$HASP890                CLASS=STC,PRIORITY=15,
$HASP890                SYSAFF=(SC64),HOLD=(NONE),
$HASP890                CMDAUTH=(LOCAL),OFFS=(),
$HASP890                SECLABEL=,USERID=STC,
$HASP890                SPOOL=(VOLUMES=(SBOX07),TGS=6,
$HASP890                PERCENT=0.0321),ARM_ELEMENT=YES,
$HASP890                CARDS=2,REBUILD=NO,SRVCLASS=,
$HASP890                SCHENV=,SCHENV_AFF=(ANY),
$HASP890                SPINNABLE=YES,CC=()
```

Figure 4-12 Operator command display of STC job

- JESLOG options for a TSO LOGON job and all jobs:
  - **$T T3001,SPIN**
  - **$T JQ9818,SPIN**

4.2.4 Migration considerations

Figure 4-13 illustrates spin processing in a mixed MAS environment. Case 1) and case 2) are described as follows:

- Job J123 has its JCL converted on system SC64. It then goes into execution on system SC63. Because system SC63 is a Release 10 system, this member cannot perform JESLOG spin processing. This would be true also for an earlier system before Release 10.
- Job J901 has its JCL converted on system SC63. Because this system is Release 10, or if it was any earlier version system, the job can never perform JESLOG spin processing—no matter which system it executes on.

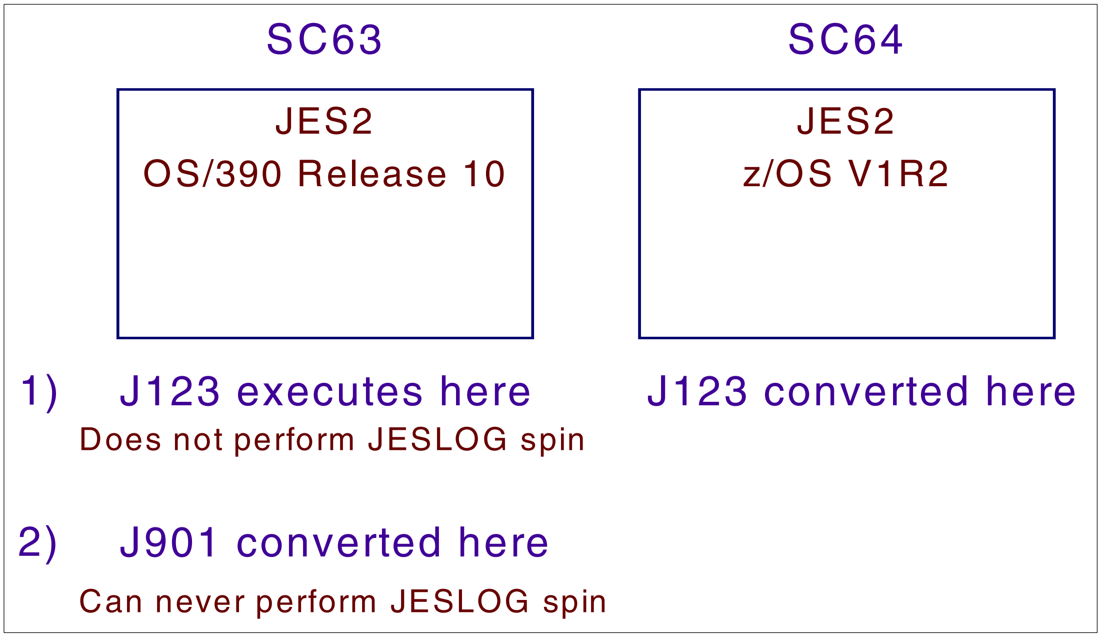


Figure 4-13 Mixed MAS environment and JESLOG spin processing

4.3 Dynamic PROCLIB support

Before this release, the only way to define or update the PROCLIB data sets for use by jobs submitted through JES2 is to place the PROCLIB concatenation in the JES2 procedure (PROC). This means that JES2 must be restarted or a re-IPL has to be done on each system in the JES2 MAS. Additionally, if the update to the JES2 procedure has a problem (or if a PROCLIB data set is deleted and the PROC is not updated), then there may be a problem restarting JES2.

4.3.1 PROCLIB initialization statement

To solve these problems, a new PROCLIB initialization statement is created and a set of commands is added to manipulate the PROCLIB data sets. The new PROCLIB statements supplement the PROCLIB DD statements in the JES2 PROC. When looking for a PROCLIB for the converter to use, first the dynamic PROCLIB statements are scanned. If no matching PROCLIB is found, then the PROCLIB in the JES2 PROC are used. Dynamic PROCLIB statements can replace, but not update or delete PROCLIBs defined in the JES2 PROC.

The PROCLIB initialization statement is used to define a logical DD name for the converter to use as follows:

- Up to 255 data set names can be associated with the PROCLIB statement.
- The data set name is required.
- The VOLSER and UNIT are only required if needed for allocation processing to complete for the data set.

The syntax for the new PROCLIB initialization statement is shown in Figure 4-14.

```
PROCLIB(dddddddd) DD(n)=(DSNAME=dsn,

                         VOLSER=vol,

                         UNIT=unit),

                         UNCONDITIONAL
```

Figure 4-14 Syntax for PROCLIB initialization statement

Where:

dddddddd Can be the same as a dd name in the JES2 PROC.

n Can be up to 255.

UNCONDITIONAL When specified, if even one of the DDs fails allocation processing, the PROCLIB statement is honored.

- Up to 255 DDs per PROCLIB.
- VOLSER and UNIT are optional (if cataloged).

PROCLIB example

Figure 4-15 on page 66shows examples of the new and old way to define procedure libraries to JES2.

```
Old way (Static PROCLIB) In JES2 PROC:

//PROC01  DD   DSN=USER.PROCLIB1,VOL=SER=J2PROC,UNIT=3390
//        DD   DSN=USER.PROCLIB2,VOL=SER=J2PROC,UNIT=3390
//        DD   DSN=SYS1.PROCLIB

New way (Dynamic PROCLIB) In JES2 initialization stream

PROCLIB(PROC01) DD(1)=(DSN=USER.PROCLIB1,VOLSER=J2PROC,UNIT=3390),
                DD(2)=(DSN=USER.PROCLIB2,VOLSER=J2PROC,UNIT=3390),
                DD(3)=(DSN=SYS1.PROCLIB)
```

Figure 4-15 Defining PROCLIBs the old and new way

When a PROCLIB is defined dynamically, it can be modified using operator commands using the **`$T PROCLIB(PROC01),DD(1)=...,DD(2)=..`** command.

4.3.2 PROCLIB statement processing

When you specify the PROCLIB statements in the initialization stream, the data sets specified in the PROCLIB statements that can be allocated will be allocated. Those statements that can not be allocated are skipped. If a DD(n) is skipped when defining a PROCLIB, then the latter DDs are moved up. There will be no “holes” in the DD(n) list.

The dynamic PROCLIB support does not use the DD name in the subscript to actually perform the allocation. Instead, it allows the system to dynamically assign a DD name to the allocation. This is generally transparent to the installation. However, the system-generated DD name may appear in some message and in a dump of the JES2 address space.

During conversion processing, JES2 opens the PROCLIB the job requested and passes the open DCB to the conversion. However, when a job completes conversion, the PROCLIB is not closed (in case the next job needs it). Because of this, the PROCLIB can remain open for extended periods of time, even though it has been deleted.

4.3.3 Operator commands

The following operator commands are available to dynamically determine the PROCLIB data sets to be used: **`$ADD PROCLIB`**, **`$DEL PROCLIB`**, **`$T PROCLIB`**, and **`$D PROCLIB`**.

$ADD PROCLIB command

The **`$ADD PROCLIB`** command works similar to the PROCLIB initialization statement. It creates a new dynamic PROCLIB concatenation, as shown in “PROCLIB initialization statement” on page 65 and in Figure 4-16. The command syntax is as follows:

```
$ADD PROCLIB(dddddddd),DD(n)=(...),
                       UNCONDITIONAL
```

```
$addproclib(testproc),dd(1)=(dsname=esilva.proclib),unconditional
$HASP319 PROCLIB(TESTPROC)  DD(1)=(DSNAME=ESILVA.PROCLIB)
```

Figure 4-16 $add proclib command example

$T PROCLIB command

The **$T PROCLIB** command can be used to update an existing dynamic PROCLIB. If the DSNAME of a DD is set to null, then that DD is removed from the concatenation. The NAME= operand is used to rename an existing dynamic PROCLIB concatenation. The command syntax is as follows:

```
$T PROCLIB(dddddddd),DD(n)=(...),
                   NAME=nnnnnnnn,
                   UNCONDITIONAL
```

Here, shown in Figure 4-17, we rename the DD name of the dynamic proclib from TESTPROC to TEMPROC.

```
$TPROCLIB(TESTPROC),dd(1)=(dsname=esilva.proclib),name=tempproc
$HASP319 PROCLIB(TEMPPROC)  DD(1)=(DSNAME=ESILVA.PROCLIB)
```

Figure 4-17 $T proclib command example

$DEL PROCLIB command

The **$DEL PROCLIB** command deletes an existing dynamic PROCLIB. It cannot affect or delete static PROCLIBs defined in the JES2 PROC. The command syntax is as follows:

```
$DEL PROCLIB(dddddddd)
               DEBUG
```

```
$delproclib(testproc)
$HASP319 PROCLIB(TESTPROC)
$HASP319 PROCLIB(TESTPROC)  DD(1)=(DSNAME=ESILVA.PROCLIB)
$HASP319                    ELEMENT DELETED
```

Figure 4-18 $DEL proclib command example

A PROCLIB is not deleted, and the data sets unallocated, until all conversion subtasks have closed the data sets. If you need to display PROCLIBs that have been deleted (via command), but have not been unallocated, use the **$D PROCLIB,DEBUG** command.

$D PROCLIB command

To display the TESTPROC and TEMPROC, use the **$D PROCLIB** command. The **$D PROCLIB** command can be used to display the current dynamic PROCLIBs. It cannot display static PROCLIBs.

```
PROCLIB that was created in Figure 4-16 on page 66
$DPROCLIB
$HASP319 PROCLIB(TESTPROC)  DD(1)=(DSNAME=ESILVA.PROCLIB)

PROCLIB that was deleted in Figure 4-18 on page 67
$DPROCLIB(TESTPROC)
$HASP003 RC=(52),
$HASP003 RC=(52),PROCLIB(TESTPROC)  - NO SELECTABLE ENTRIES
$HASP003         FOUND MATCHING SPECIFICATION

PROCLIB that was renamed in Figure 4-17 on page 67
$DPROCLIB(TEMPPROC)
$HASP319 PROCLIB(TEMPPROC)  DD(1)=(DSNAME=ESILVA.PROCLIB)
```

Figure 4-19 Examples of the $D PROCLIB command

4.3.4 Dynamic PROCLIB considerations

The dynamic PROCLIB support does not use the DD name in the subscript to actually perform the allocation. Instead, it allows the system to dynamically assign a DD name to the allocation. This is generally transparent to the installation. However, the system-generated DD name may appear in some message and in a dump of the JES2 address space.

During conversion processing, JES2 opens the PROCLIB the job requested and passes the open DCB to conversion. However, when a job completes conversion, the PROCLIB is not closed (in case the next job needs it). Because of this, the PROCLIB can remain open for extended periods of time, even though it has been deleted. A PROCLIB is not deleted, and the data sets unallocated, until all conversion subtasks have closed the data sets. If you need to display PROCLIBs that have been deleted (via command), but have not been unallocated, use the **`$D PROCLIB,DEBUG`** command.

DEBUG option

If the DEBUG option is specified, as shown in Figure 4-20, then additional information is displayed about the dynamic PROCLIB concatenation as well as logically deleted dynamic PROCLIBs that are still in use.

```
$D PROCLIB(PROC01),DEBUG
$HASP319 PROCLIB(PROC01)
$HASP319 PROCLIB(PROC01)  USECOUNT=0,DDNAME=SYS00006,
$HASP319                  CREATED=2001.149,20:42:22.36,
$HASP319                  DD(1)=(DSNAME=USER.PROCLIB1,
$HASP319                  VOLSER=J2COM1,UNIT=3390),
$HASP319                  DD(2)=(DSNAME=USER.PROCLIB2,
$HASP319                  VOLSER=J2COM1,UNIT=3390),
$HASP319                  DD(3)=(SYS1.PROCLIB)
```

Figure 4-20 Example showing the DEBUG option

Changing PROCLIB concatenations

There are several ways to change the concatenation for a dynamically created PROCLIB.

Method 1

Using the **$T** command could require several commands due to command length limitations. However, this is the easiest way if there are few data sets in the concatenation.

```
$T PROCLIB(PROC01),DD(1)=...,DD(2)=...
```

Method 2

Create a PROCLIB member using the $ADD command, as follows:

```
$ADD PROCLIB(TEMP01),DD(1)=...
```

Then use the **$T** command to add the additional concatenations as follows:

```
$T PROCLIB(TEMP01),DD(2)=...
```

You can then test the TEMP01 PROCLIB and then rename it to PROC01 as follows:

```
$T PROCLIB(TEMP01),NAME=PROC01
```

4.4 INCLUDE initialization statement

Most installations organize their initialization streams for JES2 as a series of members in a JES2 parmlib data set. The exact list of members used for an initialization is determined by a DD in the JES2 PROC. If the list of members to include needs to be changed, the JES2 PROC must be updated. If there is an error in that PROC or a data set referred to in the PROC is accidentally deleted, JES2 will not start.

With this release, a new INCLUDE initialization statement is added to process additional initialization members. This reduces the need to update the JES2 PROC. This also allows for the use of system symbolics to determine what members to include.

4.4.1 Syntax for INCLUDE statement

The syntax of the INCLUDE initialization statement specifies a required data set name, and a volser and unit (required only if needed for allocation). The data set name can have a member name. The statements in the included data set are processed immediately. When the end of the included data set is reached, processing continues with the statement after the include of the original data set. Includes can be nested. There is loop detection to prevent a nesting loop.

```
INCLUDE DSNAME=dsn,VOLSER=vol,UNIT=unit
```

The INCLUDE statement has the following considerations:

- DSNAME can include a member name.
- VOLSER and UNIT are optional (if data set is cataloged).
- Statements in the data sets are processed immediately.
- An INCLUDE data set can have INCLUDE statements.
- The current initialization deck can be displayed using the $D INCLUDE command.

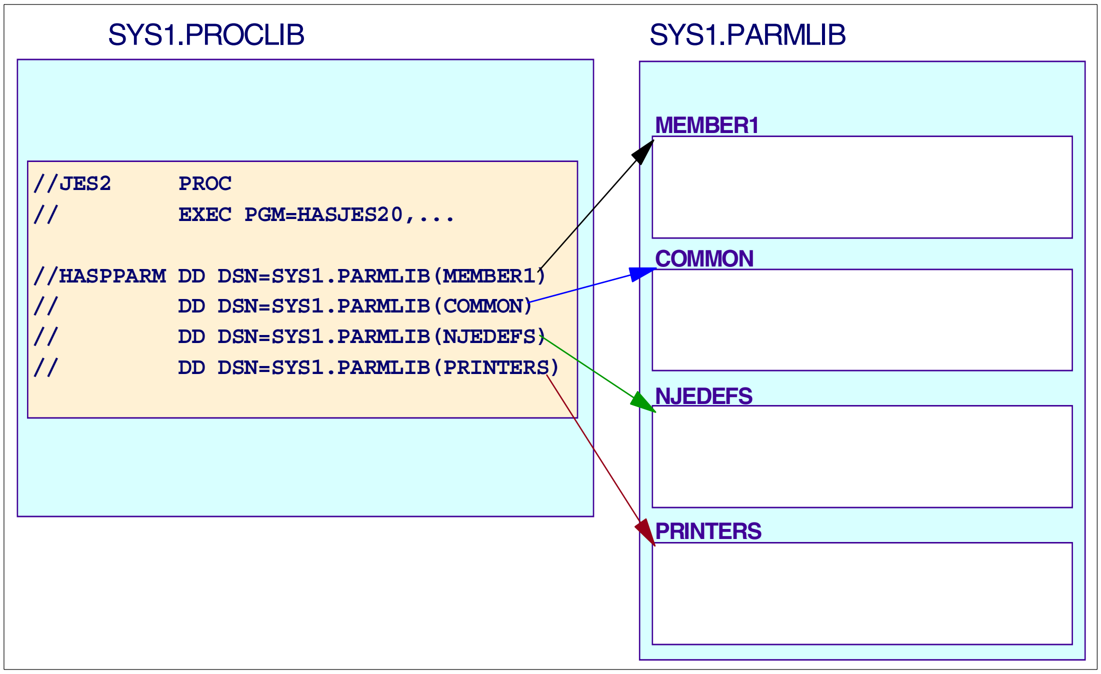
4.4.2 Implementing the INCLUDE statement

In the traditional way of setting up the JES2 PARMLIB data, the JES2 PROC points to a global member, a local member, and a member that has network data or various combinations depending on the installation. The following example shows four members that make up the JES2 initialization definitions.

```
//JES2     PROC
//         EXEC PGM=HASJES20,...
//HASPPARM DD DSN=SYS1.PARMLIB(MEMBER1)
//         DD DSN=SYS1.PARMLIB(COMMON)
//         DD DSN=SYS1.PARMLIB(NJEDEFS)
//         DD DSN=SYS1.PARMLIB(PRINTERS)
```

Figure 4-21 Sample JES2 procedure with four initialization members

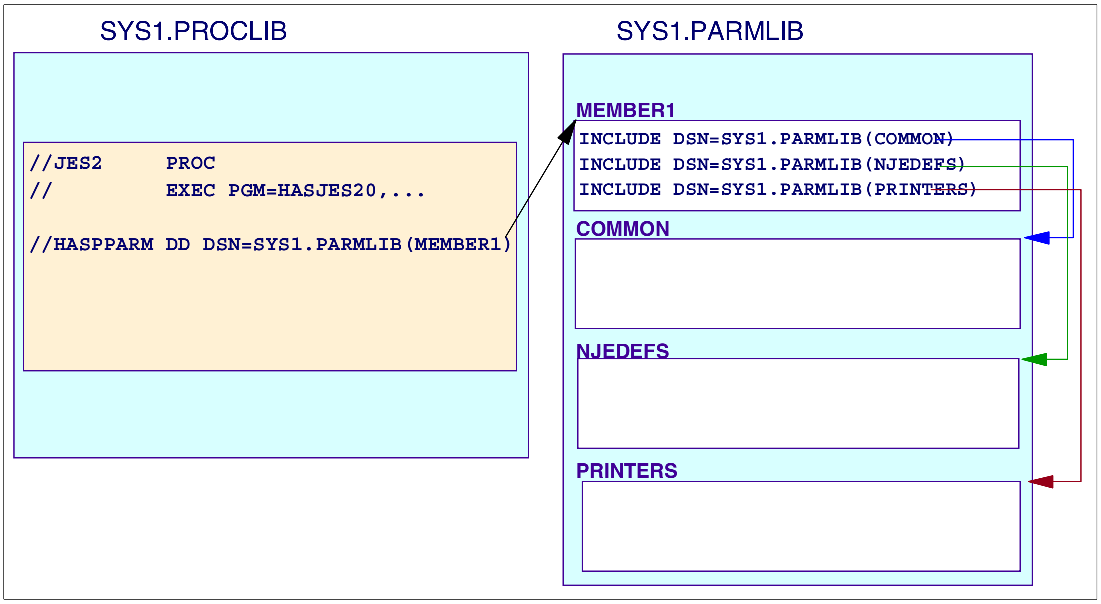
Each of the members in the HASPPARM concatenation in the procedure points to a member in SYS1.PARMLIB, as shown in Figure 4-22.



```
//JES2     PROC
//         EXEC PGM=HASJES20,...

//HASPPARM DD DSN=SYS1.PARMLIB(MEMBER1)
//         DD DSN=SYS1.PARMLIB(COMMON)
//         DD DSN=SYS1.PARMLIB(NJEDEFS)
//         DD DSN=SYS1.PARMLIB(PRINTERS)
```

Figure 4-22 HASPPARM specification in the JES2 procedure

Using the new INCLUDE statement, the JES2 PROC points to a local member, MEMBER1. MEMBER1 then INCLUDEs a common member, a network member, and a printer member. Local parameters are then processed after the INCLUDE of the common member in the local data set. Using the INCLUDE statement, any changes that need to be made to the members to process is an update to a PARMLIB member. If there is ever a problem with an INCLUDE statement, JES2 goes into console mode to resolve the error. Using the old method, an error could result in a JCL error in the JES2 PROC, preventing JES2 from starting.



```
//JES2      PROC
//          EXEC PGM=HASJES20,...

//HASPPARM DD DSN=SYS1.PARMLIB(MEMBER1)
```

```
INCLUDE DSN=SYS1.PARMLIB(COMMON)
INCLUDE DSN=SYS1.PARMLIB(NJEDEFS)
INCLUDE DSN=SYS1.PARMLIB(PRINTERS)
```

Figure 4-23 New HASPPARM specification using the INCLUDE statement

The INCLUDE statement can display the current data set that is being processed. This applies whether the current data set is the result of an INCLUDE statement or whether it was part of the HASPPARM DD statement.

4.4.3 Include statement considerations

With this release of JES2, you can use both the dynamic PROCLIB and the INCLUDE initialization statement to greatly simplify the JES2 procedure by reducing the need to update it and thereby reducing the risk of an error that would prevent JES2 from starting. A basic JES2 procedure would be:

```
//JES2     PROC
//IEFPROC EXEC PGM=HASJES20,TIME=1440,DPRTY=(15,14)
//STEPLIB   DD DSN=SYS1.LINKLIB,DISP=SHR
//HASPPARM  DD DSN=SYS1.PARMLIB(JES2PARM),DISP=SHR
```

Figure 4-24 Basic JES2 procedure

The PROCLIB and additional PARMLIB data sets can be included using the new initialization statements. In the event of a failure in the JES2 PROC, issue the following command:

```
S IEESYSAS,PROG=HASJES20,JOBNAME=JES2
```

This command will start JES2. When JES2 enters console mode, enter an INCLUDE statement to read the initial JES2 initialization deck.

4.5 Large spool support

JES2 spool volumes have more cylinders that are addressable. Since spool volumes are read by a number of vendor products (including SDSF), any change to how spool addressing operates affects these products. Specifically, JES2 supports both relative and absolute track addressing for a spool.

The current spool addressing scheme (MTTR) uses an absolute track address. Since only 2 bytes are allocated for the track portion of the address, no spool data set can cross the 64K track boundary on a volume.

By allowing the use of relative track addressing, JES2 can support a spool data set of up to 64K tracks in size anywhere on a volume. The 64K track size limit is in part due to the limits on allocating a data set extent.

Using relative track addresses requires any application that attempts to read spool (such as SDSF) to be updated to support the new rules. To limit any future impact to these programs, a new SSI interface was created to read data from spool given a track address (MTTR).

4.5.1 SPOOLDEF initialization statement

The SPOOLDEF initialization statement is updated to allow you to specify whether you "always," "as needed," or "never" want to use relative MTTRs in processing the JES2 spool. You use the new keyword, RELADDR= with a value of ALWAYS, ASNEEDED, or NEVER to specify your choice, as follows:

```
SPOOLDEF RELADDR=NEVER|ALWAYS|ASNEEDED
```

`NEVER` Never use relative addressing. Any volume that spans the 64K track boundary cannot be used.

`ALWAYS` Use relative addressing for all spool volumes. This is primarily intended as a test tool to verify that applications and exits can support relative addressing.

`ASNEEDED` Relative addressing is used if the volume crosses the 64K track boundary. (Recommended)

SPOOLDEF implementation

RELADDR applies at the time a spool volume is started. All members of the MAS must support relative addressing for it to be used.

Note: This support is rolled back to the R10 level at the end of 2001.

At the time a spool volume is started, if there are down-level members active (that do not support relative addressing), then relative addressing is not used. If no down-level members are active, relative addressing can be used (based on RELADDR). If you attempt to warm start a member that does not support relative addressing, and a volume is using relative addressing, then the member starting fails with a $HASP401 message.

The only way to reset the addressing scheme that a volume is using is to drain and restart the volume.

Operator commands

The **`$TSPOOLDEF`** command can be used to update the current setting specified in the initialization statements for RELADDR=.

If relative addressing is being used for a volume, the **`$DSPOOL,UNITDATA`** displays a new keyword, BASETRAK=. This is the value that must be added to a TT (from MTTR) to convert a relative to an absolute track address.

```
$dspool,unitdata
$HASP893 VOLUME(OP1SP1)
$HASP893 VOLUME(OP1SP1)  UNITDATA=(EXTENT=00,TRKRANGE=(003C,
$HASP893                 C3A4),RECMAX=12,BASETRAK=0040,TRKPERCYL=15)
$HASP646 0.7896 PERCENT SPOOL UTILIZATION
```

Figure 4-25 Display of spool volume showing BASETRAK

4.5.2 Spool read SSI

A new interface is provided to read records from JES2 spool. This interface is implemented as an option on the SSI 71 call. When you use this interface, you will not need to know about internal JES2 MTTR processing. This interface also provides for the use of internal JES2 device IDs that are converted to EBCDIC for use in displays by products, such as SDSF.

To simplify application access to the JES2 spool, two new functions were added to SSI 71, the JES job information service SSI. These functions allow for the reading of spool blocks and management of the storage used by these blocks. The parameter list for these functions is mapped by IAZSPLIO. The application passes the MTTR to the interface and receives as output the spool block at that MTTR. The MTTR should be treated as a token and not interpreted by the application (in the event there are future changes to the format of a spool address). Using this SSI, the application no longer has to allocate or open spool volumes to read data on them.

In addition to data records, this SSI can also read the signature records associated with a track. This is done by specifying a control block type of SIG. The MTTR passed should be the same as would be passed to $SIGIO.

Note: All of these SSI changes are documented in *Using the Subsystem Interface*.

Though the interface to access spool has been made available to applications, the format of the data returned is still subject to change from release to release of JES2.

4.6 Installation requirements

A COLD start is required to go from HJE6603 and prior releases (no support for pre-R4 mode). Therefore, you must do the following:

- Migrate to HJE6604 or higher first to avoid a cold start.

JES2 Version 1 Release 2 can coexist with the following FMIDs with APAR OW47328 needed on the downlevel systems

- HJE6607 - PTF UW99361
- HJE6608 - PTF UW99362
- HJE7703 - PTF UW99363

APAR OW47328 performs the following functions:

- Allows down-level releases to exist in a MAS with JES2 z/OS 1.2
- Adds space to the CAT to hold the JESLOG option
- Fixes problems in HASPCKPT that may be exposed by decreasing JQEs

- Adds support for the z/OS 1.2 **$TCKPTDEF,MODE=** command
- Fixes performance problem resulting in XEQ $WAITing for a job that is BERT locked
- Corrects exposure resulting from MVS EOM changes

You can use the new **$ACTIVATE** level (z2 mode) to do the following:

- Needed for some >64K jobs functions

You can switch from R4 mode to z2 mode or z2 to R4 mode via the operator command:

```
$ACTIVATE LEVEL=
```

4.6.1 Coexistence

The JES2 component of z/OS allows four consecutive FMIDs to coexist. For JES2, coexist means that each JES2 FMID can be a member in a MAS and read the same JES2 checkpoint data set. Since not every consecutive JES2 FMID matches exactly a z/OS or OS/390 release, you can have more than four consecutive z/OS releases coexist for the four JES2 FMIDs.

4.7 Other changes

The JES2 sample command translation exit 5 has been moved to SHASSAMP.

- This exit is not automatically enabled beginning in this release

JES2 now does an automatic reply to termination WTORs as follows:

- After 5 minutes, start a message every 30 seconds to remind the operator
- After 10 minutes, take default action
- A new HASP098 action message is issued, END,DUMP

This new operator command option for the **$TCKPTDEF,MODE=DUAL/DUPLEX** command does the following:

- This allows the checkpoint mode to be switched without an all member warm start
- Simplifies migrating to using the CF for the checkpoint

This release discontinued support for APPLCOPY for the following reasons:

- APPLCOPY is impractical with large CKPT.
- Specifying APPLCOPY=PRIVATE/COMMON generates errors.

5

SDSF enhancements

This chapter describes the enhancements in OS/390 Release 12 SDSF. SDSF is an optional feature of z/OS Version 1 Release 2.

This chapter describes the following new functions:

- Improvements to the Log Title Line.
- Removal of the *PROCESS statements.
- Single quote on print panel.
- UNIT support on print panel.
- Restructure of the UCLIN Jobs.
- Addition of a report class to the DA panel.
- Make Enclaves visible to operators.
- SDSF displays multiple system affinities.
- SDSF together with RMF captures enclave SRBs for DB2.
- Maintaining filter lists between SDSF sessions.
- Improving the performance of SDSF with OPERLOG is addressed.
- A FIN APAR, PQ41542, which fixes a problem with "SJ" when it is issued against a job that has USER=&xxxxx on the job card.

5.1 SDSF new functions

The enhancements for z/OS SDSF include:

- Additional sysplex-wide panels.
- New Enclaves panel.
- New spool volumes panel.
- Support of the WLM goal mode inhibitors.
- Toleration of changes being made by JES2, consisting of:
  - Greater than 64K jobs
  - Device formatter support
  - Long running jobs
  - Large DASD devices
- Support for 64-bit virtual storage
- Functions to improve systems management in the sysplex:
  - Exploitation of the logger multi-block function. This is intended to improve the performance of SDSF's browse of sysplex-wide operlog.
  - Display of OS/390 UNIX System Services data for the sysplex.
  - Filter enhancements, to simplify the control of which systems are included in SDSF's sysplex-wide displays.
  - Improvements to SDSF server function, to reduce the complexity of managing SDSF servers.
- Improved support of the sysplex environment. SDSF provides system management for sysplex-wide OS/390 UNIX data, like that for SDSF's other sysplex-wide displays.
- Minor enhancements to improve usability and satisfy customer requirements, including support for user-defined symbols with the WHEN statement in ISFPARMS, improvement to the system command extension pop-up, and the display of additional WLM enclave classifiers.

Note: MQSeries Version 2 Release 1 is required only for sysplex-wide data. If MQ is not installed or available, SDSF displays data for a single system.

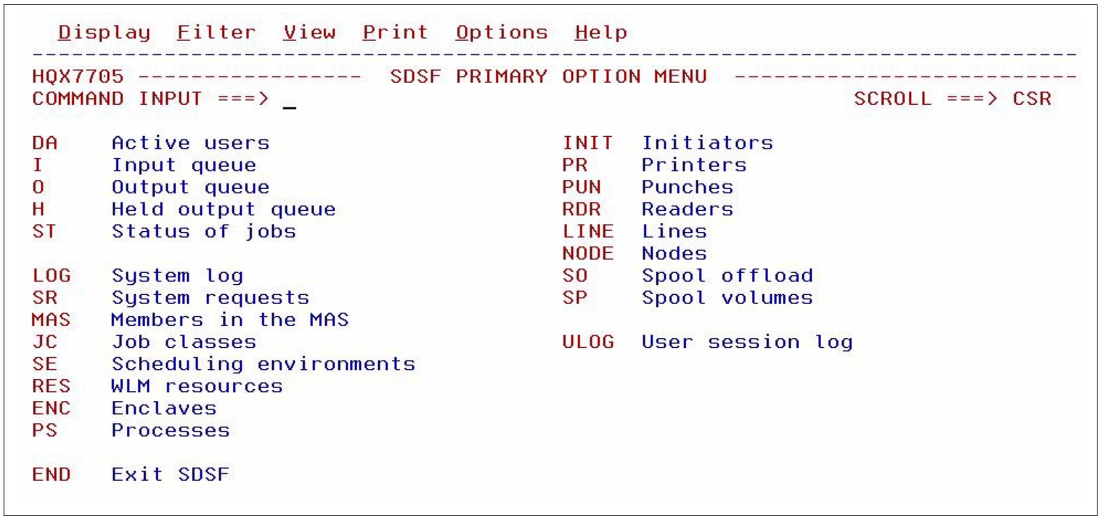
This release also fixes FIN APAR PQ41542. This fixes a problem with "`SJ`" when it is issued against a job that has USER=&xxxxx on the job card.

5.1.1 Primary Option Menu

The Primary Option Menu continues to include only those panels the user is authorized to access. There are three new options available on the menu:

- ENC - Enclaves
- PS - Processes
- SP - Spool volumes

The SDSF main menu continues to be organized by type (Job, Device, System Resources) and uses two columns to use the screen width to better advantage. The primary option menu is shown in Figure 5-1 on page 77.



```
  Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
HQX7705 -----------------  SDSF PRIMARY OPTION MENU  --------------------------
COMMAND INPUT ===> _                                          SCROLL ===> CSR

DA    Active users                      INIT  Initiators
I     Input queue                       PR    Printers
O     Output queue                      PUN   Punches
H     Held output queue                 RDR   Readers
ST    Status of jobs                    LINE  Lines
                                        NODE  Nodes
LOG   System log                        SO    Spool offload
SR    System requests                   SP    Spool volumes
MAS   Members in the MAS
JC    Job classes                       ULOG  User session log
SE    Scheduling environments
RES   WLM resources
ENC   Enclaves
PS    Processes

END   Exit SDSF
```

Figure 5-1 SDSF Primary Option Menu

5.1.2 Additional sysplex-wide panels

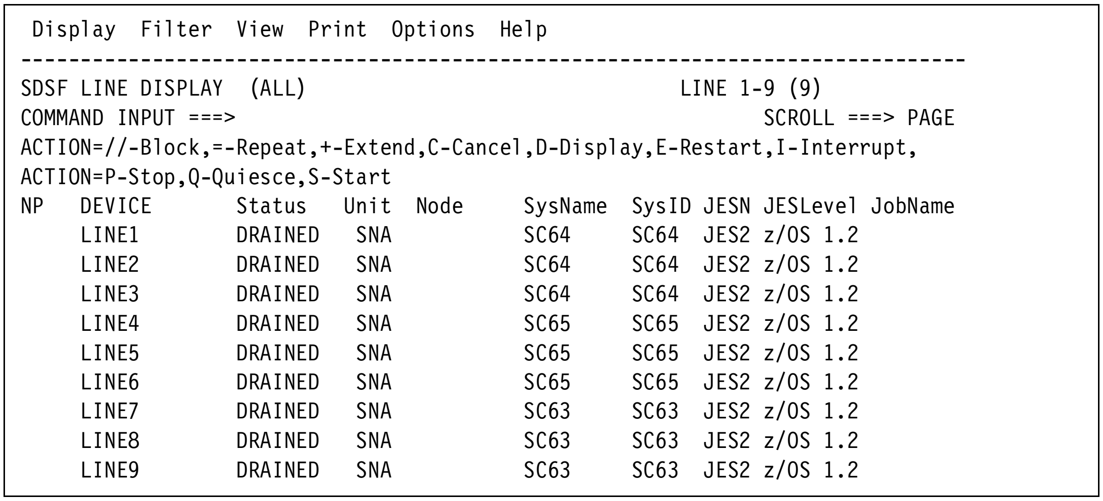
The following single-system device panels have been made sysplex-wide capable:

- Lines(LI)
- Nodes(NO)
- Punches(PUN)
- Readers(RDR)
- Spool Offload(SO)

The new columns for JESname (JESN), system name (SysName) and member name (SysID), shown in Figure 5-2 on page 78, are added to the end of the field lists by using the **Arrange** function.

Lines panel sysplex-wide

When you specify SYSNAME * to display sysplex-wide information, as on the DA panel, a value of * is displayed as (ALL). This is added to the title line of the displays and a value of blank is displayed as the name of the local system. If sysplex support is not enabled, the name of the local system is shown, in place of (ALL), regardless of the value for SYSNAME. The LI (LINES) panel display in a three system sysplex is shown in Figure 5-2.

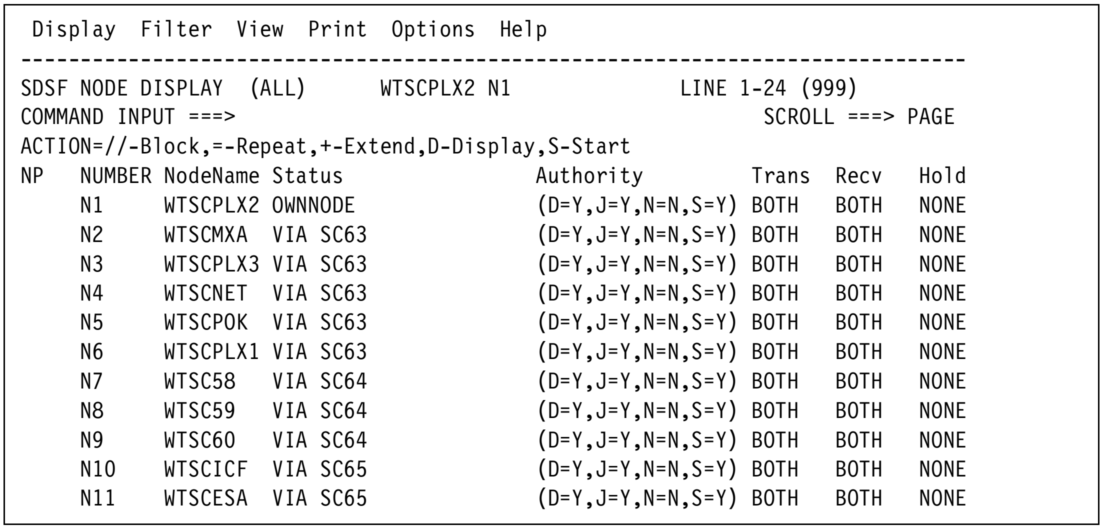


```
 Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF LINE DISPLAY  (ALL)                              LINE 1-9 (9)
COMMAND INPUT ===>                                           SCROLL ===> PAGE
ACTION=//-Block,=-Repeat,+-Extend,C-Cancel,D-Display,E-Restart,I-Interrupt,
ACTION=P-Stop,Q-Quiesce,S-Start
NP   DEVICE       Status   Unit  Node     SysName  SysID JESN JESLevel JobName
     LINE1        DRAINED   SNA           SC64     SC64  JES2 z/OS 1.2
     LINE2        DRAINED   SNA           SC64     SC64  JES2 z/OS 1.2
     LINE3        DRAINED   SNA           SC64     SC64  JES2 z/OS 1.2
     LINE4        DRAINED   SNA           SC65     SC65  JES2 z/OS 1.2
     LINE5        DRAINED   SNA           SC65     SC65  JES2 z/OS 1.2
     LINE6        DRAINED   SNA           SC65     SC65  JES2 z/OS 1.2
     LINE7        DRAINED   SNA           SC63     SC63  JES2 z/OS 1.2
     LINE8        DRAINED   SNA           SC63     SC63  JES2 z/OS 1.2
     LINE9        DRAINED   SNA           SC63     SC63  JES2 z/OS 1.2
```

Figure 5-2 Line panel display sysplex-wide

Nodes panel sysplex-wide

The title line of the NO (NODE) panel currently shows the ownnode, WTSCPLX2, and the number of the ownnode, N1, as well as (when appropriate) a PATHMGR N/A message. This message is not shown in the message area but is actually part of the title line. To make room for the SYSNAME value, (ALL), the information about the path manager is no longer displayed on the title line. Instead, it is appended to the value in the STATUS column, when appropriate. The Nodes panel is shown in Figure 5-3.

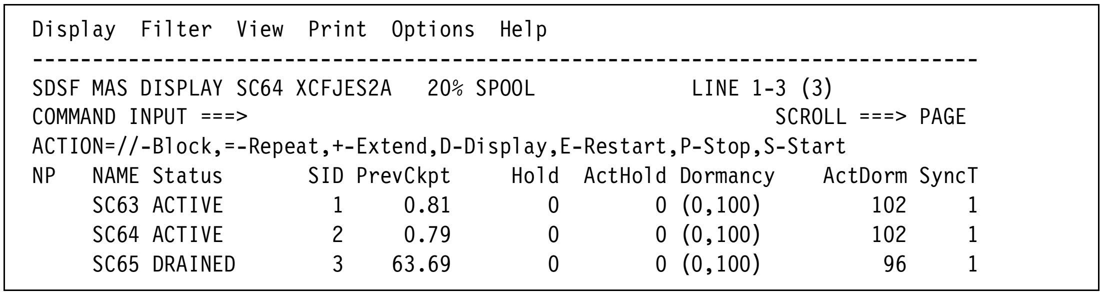


```
 Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF NODE DISPLAY  (ALL)      WTSCPLX2 N1             LINE 1-24 (999)
COMMAND INPUT ===>                                           SCROLL ===> PAGE
ACTION=//-Block,=-Repeat,+-Extend,D-Display,S-Start
NP   NUMBER NodeName Status              Authority         Trans  Recv   Hold
     N1     WTSCPLX2 OWNNODE             (D=Y,J=Y,N=N,S=Y) BOTH   BOTH   NONE
     N2     WTSCMXA  VIA SC63            (D=Y,J=Y,N=N,S=Y) BOTH   BOTH   NONE
     N3     WTSCPLX3 VIA SC63            (D=Y,J=Y,N=N,S=Y) BOTH   BOTH   NONE
     N4     WTSCNET  VIA SC63            (D=Y,J=Y,N=N,S=Y) BOTH   BOTH   NONE
     N5     WTSCPOK  VIA SC63            (D=Y,J=Y,N=N,S=Y) BOTH   BOTH   NONE
     N6     WTSCPLX1 VIA SC63            (D=Y,J=Y,N=N,S=Y) BOTH   BOTH   NONE
     N7     WTSC58   VIA SC64            (D=Y,J=Y,N=N,S=Y) BOTH   BOTH   NONE
     N8     WTSC59   VIA SC64            (D=Y,J=Y,N=N,S=Y) BOTH   BOTH   NONE
     N9     WTSC60   VIA SC64            (D=Y,J=Y,N=N,S=Y) BOTH   BOTH   NONE
     N10    WTSCICF  VIA SC65            (D=Y,J=Y,N=N,S=Y) BOTH   BOTH   NONE
     N11    WTSCESA  VIA SC65            (D=Y,J=Y,N=N,S=Y) BOTH   BOTH   NONE
```

Figure 5-3 Nodes display sysplex-wide

MAS panel display

The MAS panel previously indicated whether undefined members were included as a result of parameters on the MAS command. That information is replaced by the XCF group name, which is important because SDSFs sysplex support requires the user to be in the same MAS. The XCF group name is added to the title line of the MAS panel, as shown in Figure 5-4 on page 79.



```
 Display  Filter  View  Print  Options  Help
 -------------------------------------------------------------------------------
 SDSF MAS DISPLAY SC64 XCFJES2A   20% SPOOL            LINE 1-3 (3)
 COMMAND INPUT ===>                                           SCROLL ===> PAGE
 ACTION=//-Block,=-Repeat,+-Extend,D-Display,E-Restart,P-Stop,S-Start
 NP   NAME Status       SID PrevCkpt     Hold  ActHold Dormancy    ActDorm SyncT
      SC63 ACTIVE         1     0.81        0        0 (0,100)         102     1
      SC64 ACTIVE         2     0.79        0        0 (0,100)         102     1
      SC65 DRAINED        3    63.69        0        0 (0,100)          96     1
```

Figure 5-4 MAS display

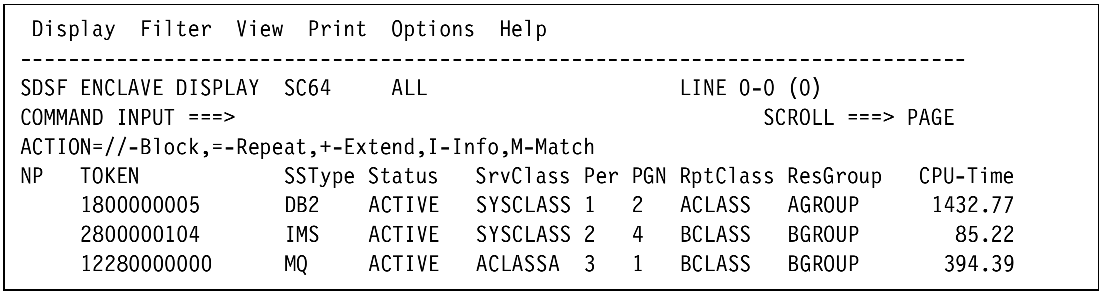
5.1.3 New enclaves panel

SDSF adds a new tabular panel that shows the properties of WLM enclaves. The panel is intended for operators. Most of the information on the panel is already provided by RMF Monitor 3. However, RMF Monitor 3 gathers its data on an interval basis rather than on a demand basis. A demand basis should be more useful for operators.

The panel shows one row per enclave per system. The panel uses MQSeries for OS/390 and SDSF servers to gather sysplex-wide data. It uses the value for SYSNAME to limit the systems displayed. No merging of data for multisystem enclaves is done, although this is being considered as a possible follow-on. Merging would show a multisystem enclave as a single row.

The fixed field for the panel is a token generated by WLM. The token is an 8-byte hexadecimal number; SDSF omits leading zeros. The token is an unusually long fixed field, but it is needed as a way to identify the enclave. Users should not have any need to type an enclave number.

By default, the panel is sorted by system, subsystem type, owner jobname, and token, as shown in Figure 5-5.



```
 Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF ENCLAVE DISPLAY  SC64     ALL                    LINE 0-0 (0)
COMMAND INPUT ===>                                           SCROLL ===> PAGE
ACTION=//-Block,=-Repeat,+-Extend,I-Info,M-Match
NP   TOKEN            SSType Status   SrvClass Per PGN RptClass ResGroup   CPU-Time
     1800000005       DB2    ACTIVE   SYSCLASS 1   2   ACLASS   AGROUP      1432.77
     2800000104       IMS    ACTIVE   SYSCLASS 2   4   BCLASS   BGROUP        85.22
     12280000000      MQ     ACTIVE   ACLASSA  3   1   BCLASS   BGROUP       394.39
```

Figure 5-5 Enclave panel display

The title line shows the setting for SYSNAME followed by ALL or ACTIVE, to indicate which ENC command is in effect. If the display is showing rows selected by the `M` action character, the MULTISYS is added to the title line.

A SCOPE column, not shown in the display because it is out to the right, indicates if the enclave is single or multi-system:

- – LOCAL indicates a local enclave, that is, one that has not been exported by WLM.
- – MULTISYS indicates a sysplex-capable enclave, meaning the enclave has been exported and has an export token.

ENC command

The **ENC** command is added; it is used to display the Enclaves panel. The syntax is as follows:

ENC (ACTIVE | ALL)

ACTIVE Displays only active enclaves

ALL Displays all enclaves, which is the default

Other commands

The **SYSNAME** command can be used to limit the systems displayed on the ENC panel.

The **QUERY AUTH** command is updated to include the ENC in the response.

The **SELECT** command is updated to filter based on the fixed field (TOKEN) of the ENC display.

The **SET ACTION** command is updated to display the **I** and **M** actions:

I This action character displays additional information about an enclave, in a pop-up.

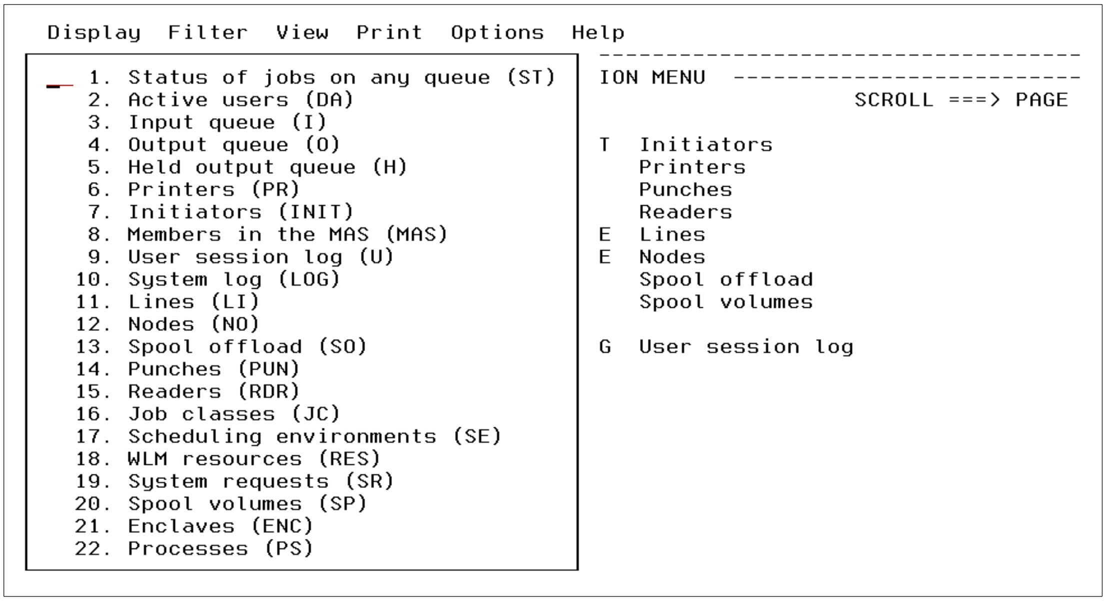
M This action character is typed next to an enclave with multisystem scope, which then displays the ENC panel with any enclaves that have an export token matching the export token for the row.

If **M** is entered for more than one row, the matches are combined. The resulting display shows enclaves with export tokens that match any of the selected rows.

When **M** is entered for an enclave with LOCAL scope, the message NOT VALID FOR TYPE is displayed.

New Display pull-down options

The Enclaves and Processes are new options 21 and 22, respectively, in the Display pull-down, as shown in Figure 5-6.

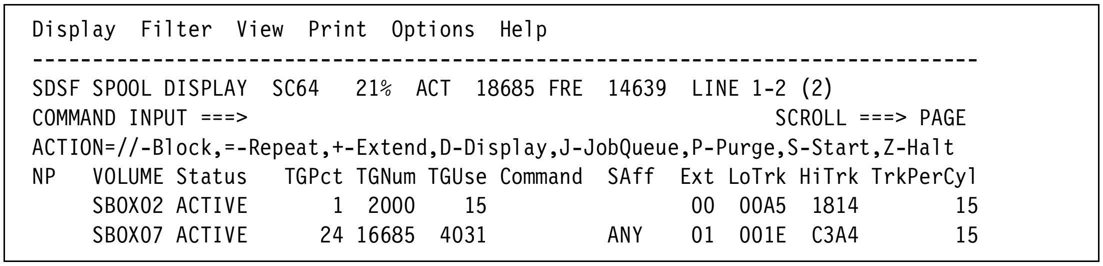


```
 Display  Filter  View  Print  Options  Help
                                             ------------------------------------
 __  1. Status of jobs on any queue (ST)     ION MENU  --------------------------
     2. Active users (DA)                                          SCROLL ===> PAGE
     3. Input queue (I)
     4. Output queue (O)                     T  Initiators
     5. Held output queue (H)                   Printers
     6. Printers (PR)                           Punches
     7. Initiators (INIT)                       Readers
     8. Members in the MAS (MAS)             E  Lines
     9. User session log (U)                 E  Nodes
    10. System log (LOG)                        Spool offload
    11. Lines (LI)                              Spool volumes
    12. Nodes (NO)
    13. Spool offload (SO)                   G  User session log
    14. Punches (PUN)
    15. Readers (RDR)
    16. Job classes (JC)
    17. Scheduling environments (SE)
    18. WLM resources (RES)
    19. System requests (SR)
    20. Spool volumes (SP)
    21. Enclaves (ENC)
    22. Processes (PS)
```

Figure 5-6 New Display pull-down options

5.1.4 Spool volumes panel

A panel is added to simplify the management of spool volumes, as shown in Figure 5-7. The panel lists statistics about each spool volume, and supports the use of action characters and overtypes to manipulate them.

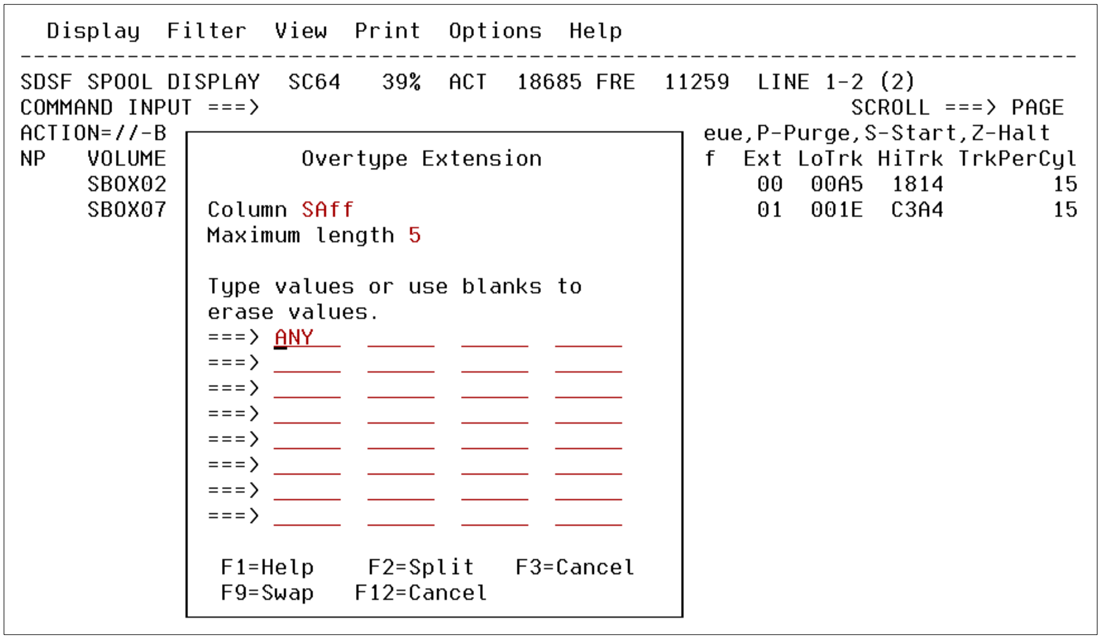


```
 Display  Filter  View  Print  Options  Help
 -------------------------------------------------------------------------------
 SDSF SPOOL DISPLAY  SC64   21%  ACT  18685 FRE  14639  LINE 1-2 (2)
 COMMAND INPUT ===>                                            SCROLL ===> PAGE
 ACTION=//-Block,=-Repeat,+-Extend,D-Display,J-JobQueue,P-Purge,S-Start,Z-Halt
 NP   VOLUME Status   TGPct TGNum TGUse Command  SAff  Ext LoTrk HiTrk TrkPerCyl
      SBOX02 ACTIVE       1  2000    15                00  00A5  1814        15
      SBOX07 ACTIVE      24 16685  4031          ANY   01  001E  C3A4        15
```

Figure 5-7   Spool volume display panel

The panel is MAS-wide. (This capability does not require MQSeries or the SDSF server.) The title line displays ACTIVE and FREE track group information.

The SP panel has one overtypeable column, SAff for system affinity. Users can overtype the field and modify a single SAff value. To work with multiple values (up to 32), type a **+** in the first column, and remove the remaining characters, to display the overtype extension pop-up, as shown in Figure 5-8.



```
  Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 SDSF SPOOL DISPLAY  SC64   39%  ACT  18685 FRE  11259  LINE 1-2 (2)
 COMMAND INPUT ===>                                            SCROLL ===> PAGE
 ACTION=//-B                                      eue,P-Purge,S-Start,Z-Halt
 NP   VOLUME         Overtype Extension           f  Ext LoTrk HiTrk TrkPerCyl
      SBOX02                                            00  00A5  1814        15
      SBOX07  Column SAff                               01  001E  C3A4        15
              Maximum length 5

              Type values or use blanks to
              erase values.
              ===> ANY___  ______  ______  ______
              ===> ______  ______  ______  ______
              ===> ______  ______  ______  ______
              ===> ______  ______  ______  ______
              ===> ______  ______  ______  ______
              ===> ______  ______  ______  ______
              ===> ______  ______  ______  ______
              ===> ______  ______  ______  ______

               F1=Help    F2=Split   F3=Cancel
               F9=Swap   F12=Cancel
```

Figure 5-8   Spool Volume pop-up

The action characters that are implemented to manipulate volumes on the SP panel are shown in Table 5-1 on page 82.

Table 5-1 Action characters for the SP panel

Character	Description	JES2 Command	Confirm
D, Dl	Display spool; Display long	$dspl	No
P,PC	Drain; Drain and Cancel jobs	$pspl	Yes
S	Start	$sspl	No
Z	Halt	$zspl	Yes

5.2 Support for WLM goal mode inhibitors

The objective of the new special protection options is to remove the major problems in WLM that prevent wider adoption by installations. This is done by providing new externals to allow performance administrators to:

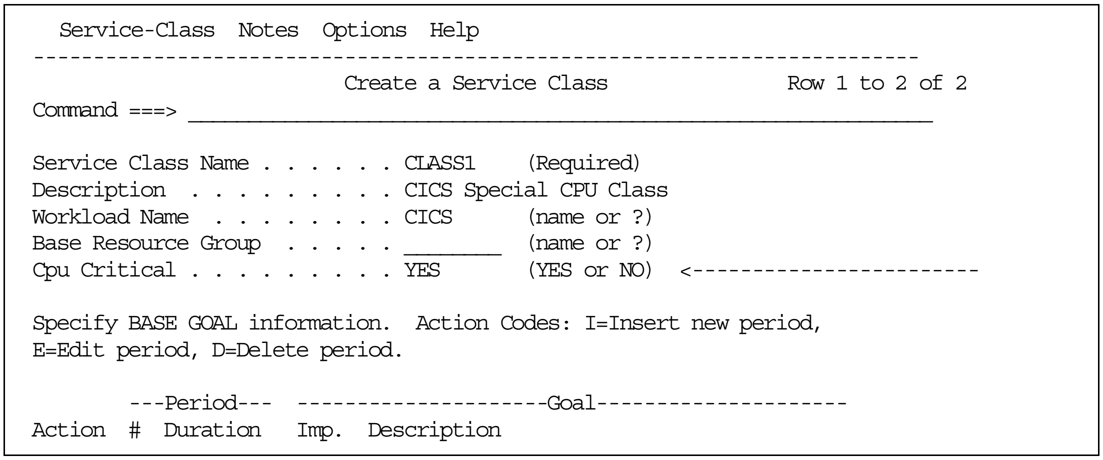
1. Protect CPU for critical regions more stringently by assigning CPU protection to assure that important address spaces always have a higher CPU dispatching priority than work of lower importance. (In this case, "importance" refers to the importance level of the goal assigned to the address space.) This provides a guarantee that CPU access for important address spaces is never impacted by work of lower importance.
2. Protect storage in a storage-constrained system. When storage is constrained, an online region that is idle for a long period of time may be exposed to paging delays when it becomes active again. The cause of this behavior is that, while the region is idle, WLM believes it requires little storage to meet its goal. When the region becomes active, WLM must relearn its storage requirements.
3. Restrict which regions are managed using transaction goals, and assign performance goals to work based on the run-time environment. Some CICS or IMS regions cannot be effectively managed using transaction response times.

However, once any CICS or IMS regions are switched to transaction response time goal management, *all* regions must be managed the same way, regardless of whether they are used in a production environment or for testing purposes.

5.2.1 CPU protection

In WLM goal mode, there is no guarantee that the dispatching priority of critical address spaces will be higher than that of the other work in the system. If the critical work is meeting its goals and there is less important work that is missing its goal, WLM may assign a higher dispatching priority to the lower importance work, as long as it does not impact the higher importance work.

APAR OW43855 provides the internals to support the CPU CRITICAL = YES option in the WLM service class definition panel, as shown in Figure 5-9 on page 83. With this support, regions or transactions with very stringent service level agreements can be marked CPU CRITICAL so that lower importance work will not have a higher dispatching priority.



```
  Service-Class  Notes  Options  Help
 --------------------------------------------------------------------------
                          Create a Service Class            Row 1 to 2 of 2
 Command ===> ______________________________________________________________

 Service Class Name . . . . . . CLASS1    (Required)
 Description  . . . . . . . . . CICS Special CPU Class
 Workload Name  . . . . . . . . CICS      (name or ?)
 Base Resource Group  . . . . . ________  (name or ?)
 Cpu Critical . . . . . . . . . YES       (YES or NO)  <------------------------

 Specify BASE GOAL information.  Action Codes: I=Insert new period,
 E=Edit period, D=Delete period.

        ---Period---  ---------------------Goal---------------------
 Action  #  Duration   Imp.  Description
```

Figure 5-9 Defining a service class as CPU critical

New service class attributes

CPU protection is part of the service class definition. Earlier designs treated CPU protection like storage protection; while this clustered all of the new externals in one place, there was no way to hide the SRM period-level CPU management. This led to SRM having to propagate CPU protection from the address spaces to periods.

CPU protection was moved to the service class definition (with WLM restricting it to single-period service classes to satisfy SRM requirements) to make its specification and SRM implementation scope consistent. Specify Yes or No in the field on the service class definition panel, as shown in Figure 5-9.

CPU protection considerations

CPU protection is always defined for a service class, never in classification rules. The following considerations should be understood:

- WLM is responsible for restricting CPU protection to single-period service classes.The decision to restrict CPU protection to single-period service classes follows from the SRM implementation of CPU management, which is period-level, and from SRM implementation restrictions about the order of dispatching priorities within multi-period service classes. This restriction does not adversely affect the primary targets of this support (CICS and IMS regions), which typically run in single-period service classes.
- Because SRM may ignore protection in certain cases, reporting interfaces that work at the address space level should distinguish between classification data and how the address space is being managed at the instant the interface is called. On performance-sensitive interfaces, it is acceptable to report only the instantaneous data.
- Reporting interfaces that provide total mode period-level data rather than address space level data should report only the specification information such as whether CPU protection was assigned.

5.2.2 Storage protection

APAR OW43810 implements a new option called Storage Critical that can be specified in the goal mode classification rules. Storage Critical=YES means that the address space keeps close to its high water mark of storage used even if storage is not needed to meet its goals.

Storage protection may be assigned to a CICS or IMS transaction service class via the classification rules, or to individual address spaces via their classification rules.

If a CICS or IMS transaction is marked as storage critical, all regions serving the transaction's service class receive protection.

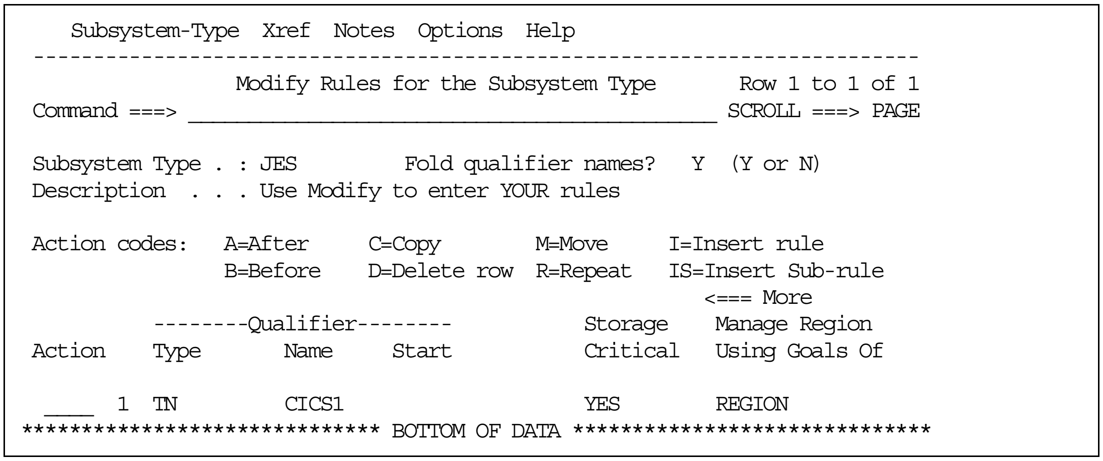
The classification rule column for storage protection should only be displayed for subsystem types CICS, IMS, ASCH, JES, STC, OMVS, TSO. While only CICS, IMS, JES, and STC are required to support the CICS or IMS requirements, there are known cases where OMVS daemons might benefit from storage protection.

Assigning storage protection using classification rules

CICS or IMS regions may be started by a job to JES or via a started task (STC). Therefore, the only two subsystem types in the WLM classification rules that support this new option are JES and STC.

When defining classification rules to assign a CICS or IMS region a service class, scrolling right by pressing PF11 twice reveals the Storage Critical and Manage Region Using Goals Of columns, as shown in Figure 5-10.

> **Note:** You can assign storage protection to all types of address spaces using classification rules for subsystem types ASCH, JES, OMVS, STC, and TSO.



```
   Subsystem-Type  Xref  Notes  Options  Help
 --------------------------------------------------------------------------
                 Modify Rules for the Subsystem Type       Row 1 to 1 of 1
 Command ===> ____________________________________________ SCROLL ===> PAGE

 Subsystem Type . : JES        Fold qualifier names?   Y  (Y or N)
 Description  . . . Use Modify to enter YOUR rules

 Action codes:   A=After     C=Copy        M=Move     I=Insert rule
                 B=Before    D=Delete row  R=Repeat   IS=Insert Sub-rule
                                                         <=== More
           --------Qualifier--------           Storage    Manage Region
 Action    Type       Name     Start           Critical   Using Goals Of

  ____   1  TN         CICS1                   YES        REGION
****************************** BOTTOM OF DATA ******************************
```

Figure 5-10   Defining storage critical in the JES classification rules

Storage protection considerations

For CICS or IMS regions, this allows performance administrators to do the following at an address space level:

- Assign long-term storage protection, so that once storage is acquired, WLM restricts storage donations, even if the region appears to have stopped serving transactions. A storage-protected address space is allowed to donate storage only in the following cases:
  - Higher importance work needs the storage to meet goals.
  - Work of equal importance needs the storage, is missing goals, and the protected address space is overachieving its goal. The protected address space can donate storage until it just meets goals.
  - Work of equal importance needs the storage, is missing goals, and the protected address space is missing its goal by less than the receiver. The protected address space can donate storage until the performance indices are equalized.

– Basically, the previous cases say that we continue to use the same donor/receiver order when dealing with importances at or above the protected address space, and prevent donations by protected address spaces to lower importance work.

For installations converting from compatibility mode, this solves an aspect of the problems cited with the removal of storage isolation, as well as sparse arrivals.

Storage protection may be specified in conjunction with:

- A short response time goal of twenty seconds or more
- A single service class
- All goals except a discretionary goal

5.2.3 SDSF support

SDSF supports new WLM goal mode inhibitors that were introduced in OS/390 Release 10. This is done by adding new columns to the DA panel and adding a new value to the existing SERVER column, as shown in Figure 5-11 on page 86. These new columns are added to the end of the DA column list, but can be moved by using the **Arrange** command, or by changing the ISFPARMS. The new columns are defined as follows:

CPUCrit This column indicates whether the address space is CPU protected or “critical” in WLM terminology.

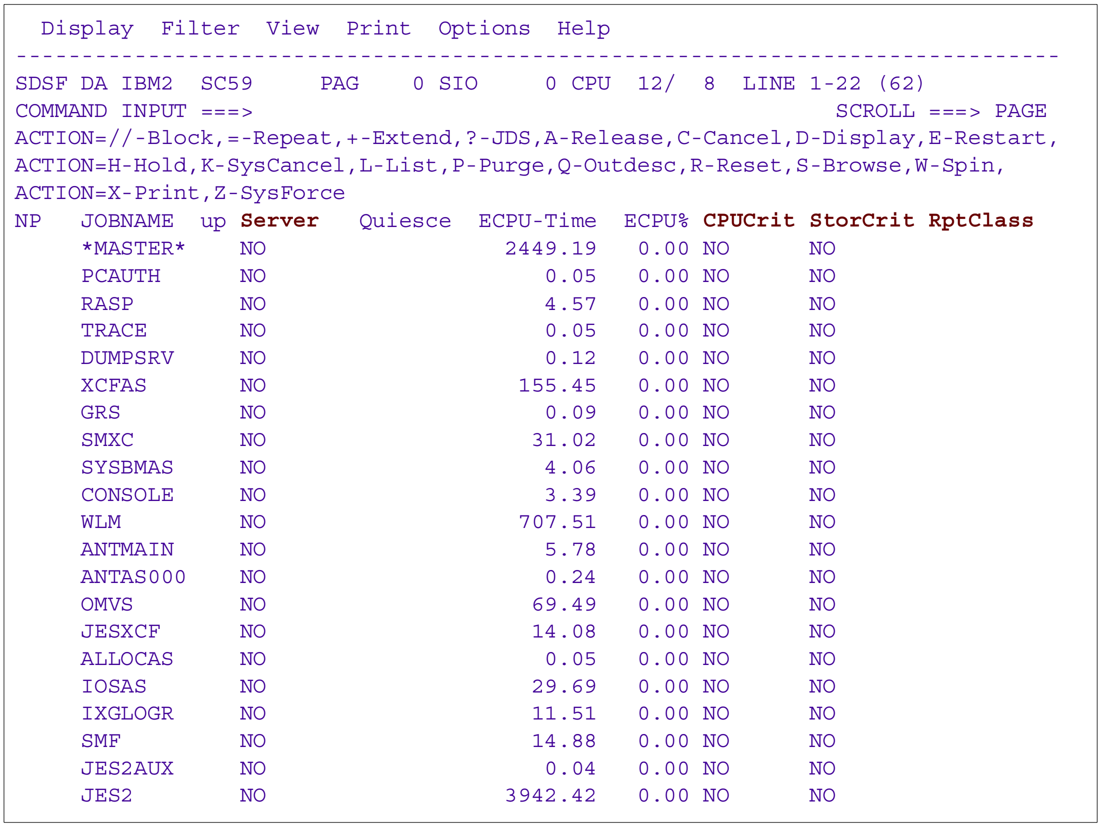
StorCrit This column indicates whether the address space is storage protected or “critical” in WLM terminology.

RptClass This column is added to display the report class.

Server The existing server column has a new value, N/A. N/A is displayed if the address space is exempt from server management.

Note: The columns are displayed if at least one system in the sysplex is in goal mode and the values for these columns are displayed as YES or NO.

It is also possible to have a mixed sysplex, in which some systems are not at the Release 10 level. For environments that are not appropriate, the system is not in goal mode or running at least Release 10, the CPUCrit and StorCrit columns display `NO`, and RptClass is blank.



```
  Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF DA IBM2  SC59     PAG    0 SIO     0 CPU  12/  8  LINE 1-22 (62)
COMMAND INPUT ===>                                            SCROLL ===> PAGE
ACTION=//-Block,=-Repeat,+-Extend,?-JDS,A-Release,C-Cancel,D-Display,E-Restart,
ACTION=H-Hold,K-SysCancel,L-List,P-Purge,Q-Outdesc,R-Reset,S-Browse,W-Spin,
ACTION=X-Print,Z-SysForce
NP   JOBNAME  up Server   Quiesce  ECPU-Time  ECPU% CPUCrit StorCrit RptClass
     *MASTER*    NO                  2449.19   0.00 NO      NO
     PCAUTH      NO                     0.05   0.00 NO      NO
     RASP        NO                     4.57   0.00 NO      NO
     TRACE       NO                     0.05   0.00 NO      NO
     DUMPSRV     NO                     0.12   0.00 NO      NO
     XCFAS       NO                   155.45   0.00 NO      NO
     GRS         NO                     0.09   0.00 NO      NO
     SMXC        NO                    31.02   0.00 NO      NO
     SYSBMAS     NO                     4.06   0.00 NO      NO
     CONSOLE     NO                     3.39   0.00 NO      NO
     WLM         NO                   707.51   0.00 NO      NO
     ANTMAIN     NO                     5.78   0.00 NO      NO
     ANTAS000    NO                     0.24   0.00 NO      NO
     OMVS        NO                    69.49   0.00 NO      NO
     JESXCF      NO                    14.08   0.00 NO      NO
     ALLOCAS     NO                     0.05   0.00 NO      NO
     IOSAS       NO                    29.69   0.00 NO      NO
     IXGLOGR     NO                    11.51   0.00 NO      NO
     SMF         NO                    14.88   0.00 NO      NO
     JES2AUX     NO                     0.04   0.00 NO      NO
     JES2        NO                  3942.42   0.00 NO      NO
```

Figure 5-11   DA display panel showing new WLM columns

5.3 SDSF support for JES2 changes

Some changes made to JES2 in z/OS Version 1 Release 2 are supported by changes to SDSF in this release. The support for the functional changes introduced in JES2 are as follows:

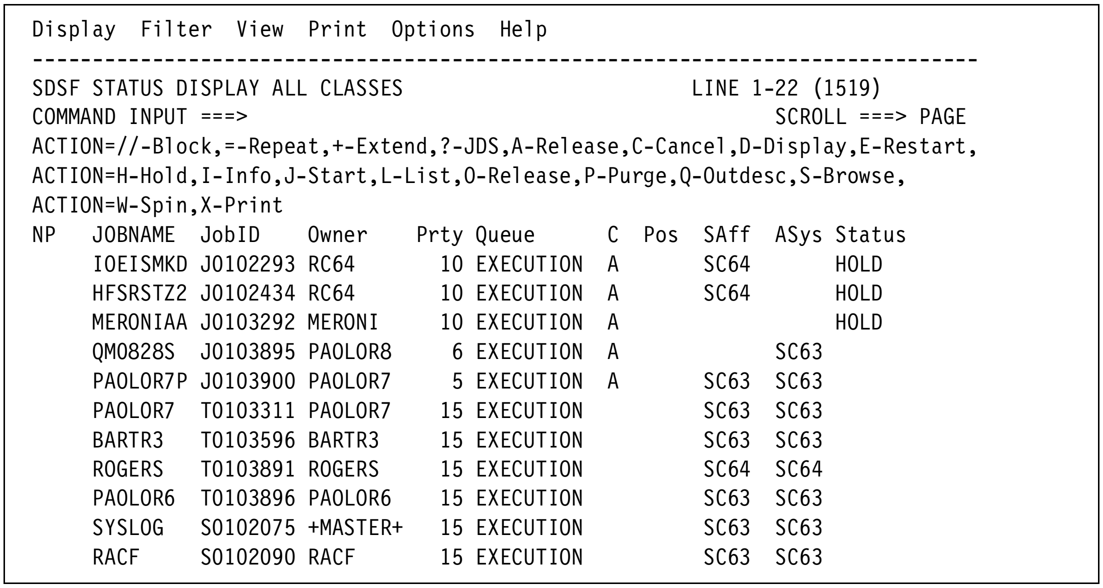
- Greater than 64K jobs.
- Better management of JES log data sets for long running jobs.
- Spool now supports data sets up to 64K tracks placed anywhere on a volume.

5.3.1 Greater than 64K jobs

SDSF adds toleration of support in OS/390 R12 JES2 for greater than 64K jobs in the MAS, as well as for high job numbers. This affects the display of jobids in columns and panel titles. It also affects the SELECT command, which accepts jobid as a parameter, and SAF resources.

Changes made to SDSF panels

To support the new job number range of 1 to 999999, JES2 displays jobids using an abbreviated alphabetic prefix (J, T, or S in place of JOB, TSU and STC) when there are more than 64K jobs, as shown in Figure 5-12 on page 87. When SDSF receives the jobid to display in a column, it is displayed in a format consistent with JES2 messages.



```
Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF STATUS DISPLAY ALL CLASSES                        LINE 1-22 (1519)
COMMAND INPUT ===>                                            SCROLL ===> PAGE
ACTION=//-Block,=-Repeat,+-Extend,?-JDS,A-Release,C-Cancel,D-Display,E-Restart,
ACTION=H-Hold,I-Info,J-Start,L-List,O-Release,P-Purge,Q-Outdesc,S-Browse,
ACTION=W-Spin,X-Print
NP   JOBNAME  JobID    Owner    Prty Queue      C  Pos  SAff  ASys Status
     IOEISMKD J0102293 RC64       10 EXECUTION  A       SC64       HOLD
     HFSRSTZ2 J0102434 RC64       10 EXECUTION  A       SC64       HOLD
     MERONIAA J0103292 MERONI     10 EXECUTION  A                  HOLD
     QM0828S  J0103895 PAOLOR8     6 EXECUTION  A             SC63
     PAOLOR7P J0103900 PAOLOR7     5 EXECUTION  A       SC63  SC63
     PAOLOR7  T0103311 PAOLOR7    15 EXECUTION          SC63  SC63
     BARTR3   T0103596 BARTR3     15 EXECUTION          SC63  SC63
     ROGERS   T0103891 ROGERS     15 EXECUTION          SC64  SC64
     PAOLOR6  T0103896 PAOLOR6    15 EXECUTION          SC63  SC63
     SYSLOG   S0102075 +MASTER+   15 EXECUTION          SC63  SC63
     RACF     S0102090 RACF       15 EXECUTION          SC63  SC63
```

Figure 5-12 Display of new JOBIDs

New column widths

On the following SDSF panels, DA, I, H, INIT, LI, PR, PUN, RDR, SO, and ST, the default width of the JNum column is changed from 5 to 6.

ISFPARMS customization

Installations with their own fields lists in ISFPARMS may want to make the change to column widths.

When running SDSF R12, APPLCOPY is no longer supported. SDSF automatically uses VERSIONS instead. The JESDATA keyword in ISFPARMS is now ignored in R12, but there are no messages issued.

5.3.2 Long-running jobs

To support the new implementation of spinning of the job and message logs for long-running jobs, SDSF adds toleration support in OS/390 R12. JES2 has added options to spin the joblog and message logs, or to suppress messages from being written to the log while the job is active. The job and message logs can now be viewed in logical order. This new SDSF support is similar to the SDSF support for editing JCL with the **SJ** action character.

SDSF has added new columns to display and modify the joblog option. New action characters are added to view the logs in logical order, and an additional browse option shows the logs in logical order. An action character is also added to cause the job and message logs to spin, and a **DL** action character is added to display options not shown with D.

New SDSF columns

A new column is added to the end of the field lists for the DA, I and ST panels. Table 5-2 describes the SPIN column. JES2 displays the SPINNABLE attribute only if the job is in execution. The column shows `YES` or `NO` for jobs in execution, and is blank, otherwise. On DA, Spin is available only with RMF.

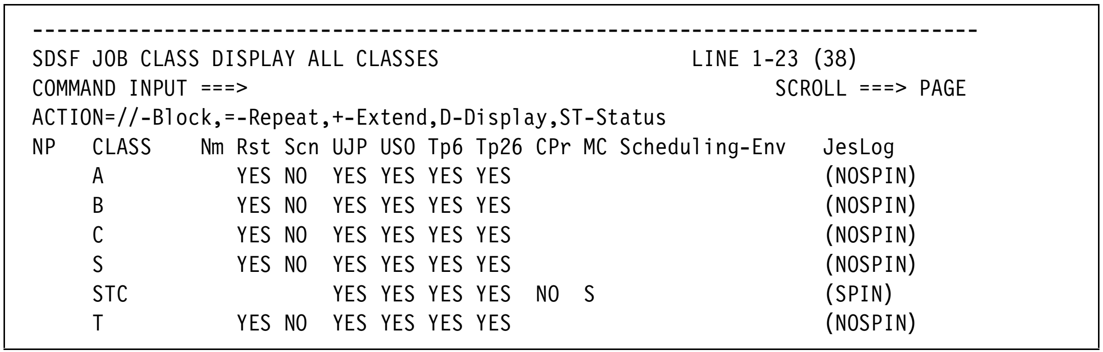
Table 5-2 New column on DA, I and ST Panels

Column (SDSF name)	Title (displayed)	Width	Description	Access immed	Access delay
SPIN	Spin	4	Indicator of whether jobs in the job class can be spun		Yes

A JesLog column is also added to the end of the field lists for the JC panel, as shown in Table 5-3 and displayed with SDSF as shown in Figure 5-13. You can overtype this field and set the spin option for the job class. It has the same values as returned by a **`$DJOBCLASS`** command, such as (SPIN,100K), SUPPRESS, or NOSPIN.

Table 5-3 New column on JC Panel

Column (SDSF name)	Title (displayed)	Width	Description	Access immed	Access delay
JESLOG	JesLog	13	Spin options for JES2 joblogs	Yes	



```
-------------------------------------------------------------------------------
SDSF JOB CLASS DISPLAY ALL CLASSES                     LINE 1-23 (38)
COMMAND INPUT ===>                                            SCROLL ===> PAGE
ACTION=//-Block,=-Repeat,+-Extend,D-Display,ST-Status
NP   CLASS    Nm Rst Scn UJP USO Tp6 Tp26 CPr MC Scheduling-Env   JesLog
     A           YES NO  YES YES YES YES                          (NOSPIN)
     B           YES NO  YES YES YES YES                          (NOSPIN)
     C           YES NO  YES YES YES YES                          (NOSPIN)
     S           YES NO  YES YES YES YES                          (NOSPIN)
     STC                 YES YES YES YES  NO  S                   (SPIN)
     T           YES NO  YES YES YES YES                          (NOSPIN)
```

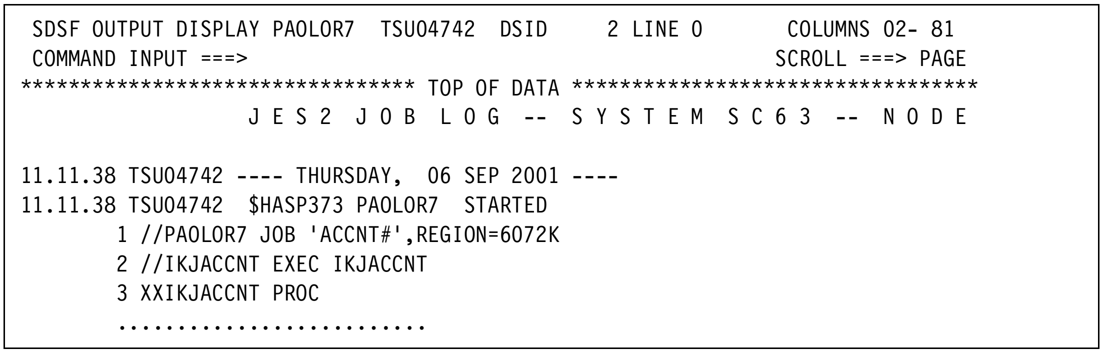
Figure 5-13 Job class (JC) panel showing JesLog column

Users can change the location or width of the columns with the Arrange function. System programmers can change the column width or title for a group of users with the ISFFLD macro or FLD statement in ISFPARMS.

Action characters

The S, SB, and SE action characters now logically concatenate job logs and message logs when there are spun data sets as follows:

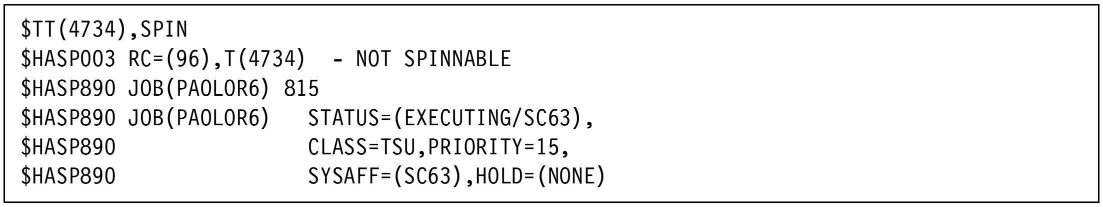
S This action character is changed to display joblogs and message logs in chronological order as closely as possible. From the JDS action character, it shows just the single data set that is selected. SB places you into ISPF browse mode. SE places you into ISPF edit mode.



```
 SDSF OUTPUT DISPLAY PAOLOR7  TSU04742  DSID     2 LINE 0       COLUMNS 02- 81
 COMMAND INPUT ===>                                            SCROLL ===> PAGE
********************************* TOP OF DATA **********************************
                  J E S 2  J O B  L O G  --  S Y S T E M  S C 6 3  --  N O D E

11.11.38 TSU04742 ---- THURSDAY,  06 SEP 2001 ----
11.11.38 TSU04742  $HASP373 PAOLOR7  STARTED
        1 //PAOLOR7 JOB 'ACCNT#',REGION=6072K
        2 //IKJACCNT EXEC IKJACCNT
        3 XXIKJACCNT PROC
        ..........................
```

Figure 5-14 Display of job log using the S action character

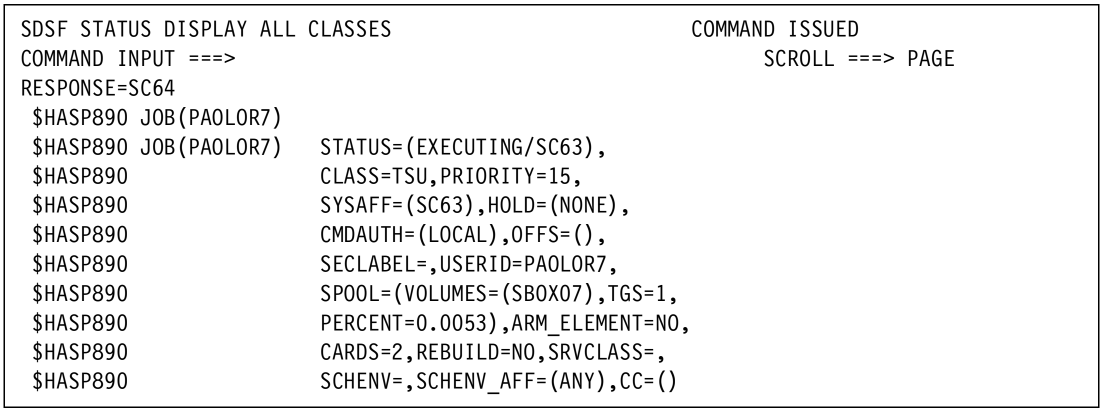
W This action character is added to DA, I, and ST panels and it causes the job and message logs to spin. It is valid only if the job is in a class for which spinning is allowed, and requires RMF. You must look at the log to see the response, as shown in Figure 5-15.



```
$TT(4734),SPIN
$HASP003 RC=(96),T(4734)  - NOT SPINNABLE
$HASP890 JOB(PAOLOR6) 815
$HASP890 JOB(PAOLOR6)   STATUS=(EXECUTING/SC63),
$HASP890                CLASS=TSU,PRIORITY=15,
$HASP890                SYSAFF=(SC63),HOLD=(NONE)
```

Figure 5-15 Response to W action character from the syslog

DL This form of the D action character is added to DA, I, and ST panels. It issues a **$D, LONG** command, which shows options that are not displayed with the **$D** command issued by the D action character.



```
SDSF STATUS DISPLAY ALL CLASSES                        COMMAND ISSUED
COMMAND INPUT ===>                                           SCROLL ===> PAGE
RESPONSE=SC64
 $HASP890 JOB(PAOLOR7)
 $HASP890 JOB(PAOLOR7)   STATUS=(EXECUTING/SC63),
 $HASP890                CLASS=TSU,PRIORITY=15,
 $HASP890                SYSAFF=(SC63),HOLD=(NONE),
 $HASP890                CMDAUTH=(LOCAL),OFFS=(),
 $HASP890                SECLABEL=,USERID=PAOLOR7,
 $HASP890                SPOOL=(VOLUMES=(SBOX07),TGS=1,
 $HASP890                PERCENT=0.0053),ARM_ELEMENT=NO,
 $HASP890                CARDS=2,REBUILD=NO,SRVCLASS=,
 $HASP890                SCHENV=,SCHENV_AFF=(ANY),CC=()
```

Figure 5-16 Response on SDSF screen to the DL action character

5.4 Support for 64-bit virtual storage

z/OS Version 1 Release 2 introduces the initial support for 64-bit virtual addressing. Programs can access the virtual storage area above the 2 GB address. The basic 64-bit virtual storage support introduced in this release is the foundation for the 64-bit z/OS operating system virtual storage infrastructure. SDSF adds support for the 64-bit virtual function, which consists of:

- A new column on the DA display for the memlimit, using data provided by RMF.
- SDSF shows scaled numeric values using T, M, and B, for thousands, millions, and billions.
- Changes to the Filter routine to allow filtering on the large integer.

Note that for other numeric columns, SDSF has a maximum valid value for input of 99999999.

New MemLimit column

The value for the MemLimit column is formatted using a new service that uses K, M, G, T and P for scaling. To avoid confusion between the different uses of M and T in different columns, SDSF adds scaling for binary columns that uses the abbreviations KB, MB, GB, TB and PB. These new abbreviations are documented in the product documentation and on-line help. The MemLimit column is added to the end of the field list for the DA panel, as described in Table 5-4.

Table 5-4 New column on DA Panel

Column (SDSF name)	Title (displayed)	Width	Description	Access immed	Access delay
MEMLIMIT	MemLimit	8	Memory limit (z/Architecture only)	Yes	

The MemLimit value can be filtered using the existing filter externals. The input is restricted to values supported where the largest non-scaled number accepted as input is 2147483649. The SDSF support allows you to:

- Change the width of the MemLimit column to 10 or more.
- Enter a value with an abbreviation, such as MB. For example, rather than entering “filter memlimit ge 4294967295,” which is invalid, you can enter “filter memlimit ge 4095MB”.
- Accept as input a number, or a number with any of the abbreviations used in that column (KB, MB,GB, TB, and PB). For example, 4096 and 4MB are both valid when entering a filter, although SDSF always displays that value in the MemLimit column as 4MB.

Note: Using greater than or less than comparisons, rather than equals, is recommended with this column.

SDSF and MemLimit field

SDSF does not accept a number with an abbreviation in any columns that use the other type of scaling, that is, where the abbreviations are T, M, and B.

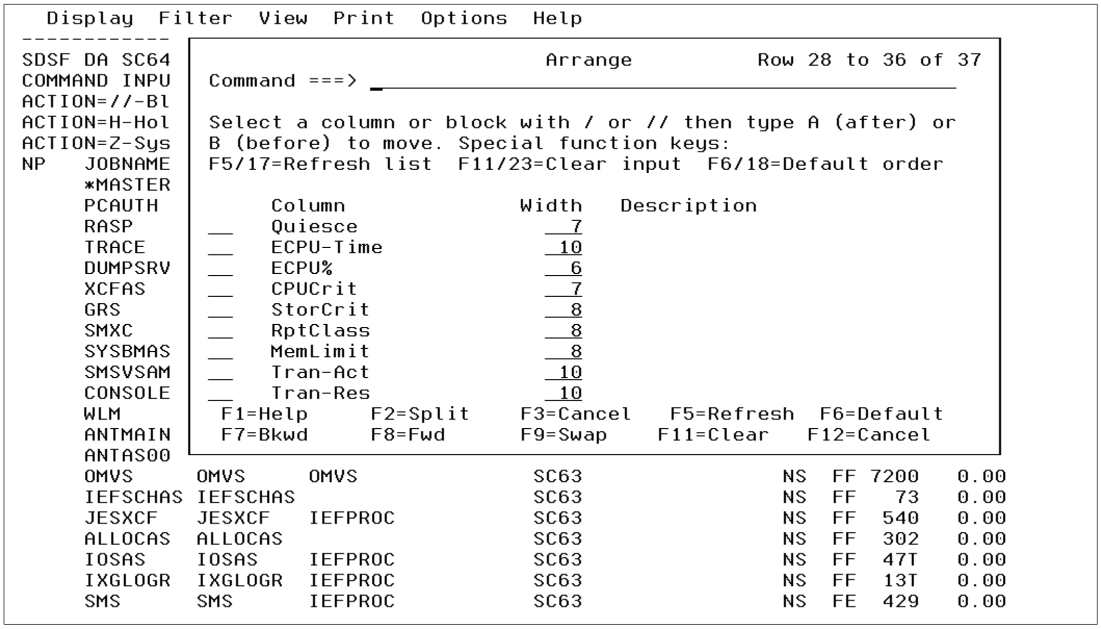
- If the value fits within the field width, it is displayed as digits (1 to 20) or, when the value is an exact multiple of K,M,G,T or P, as the number plus the appropriate abbreviation. For example, 1024 is displayed as 1KB rather than 1024.
- If the value does not fit within the field width, SDSF tries to use a short format. For the short format, the service truncates the value to have a maximum length of 6. This

approach is different than SDSF scaling, which tries to get the best fit for the field width. The service uses 6 as the maximum length in all cases.

- If the value cannot be shown in short mode, asterisks are displayed in the field.
- If the value is 0, the field is blank.

Displaying the MemLimit column

Users can change the location or width of the column with the Arrange function, as shown in Figure 5-17. System programmers can change the column width or heading for a group of users with the ISFFLD macro or FLD statement in ISFPARMS.



```
  Display  Filter  View  Print  Options  Help
------------
SDSF DA SC64 |                           Arrange         Row 28 to 36 of 37 |
COMMAND INPU | Command ===> _                                               |
ACTION=//-Bl |                                                              |
ACTION=H-Hol | Select a column or block with / or // then type A (after) or |
ACTION=Z-Sys | B (before) to move. Special function keys:                   |
NP   JOBNAME | F5/17=Refresh list  F11/23=Clear input  F6/18=Default order  |
     *MASTER |                                                              |
     PCAUTH  |      Column             Width   Description                  |
     RASP    | __   Quiesce               7                                 |
     TRACE   | __   ECPU-Time            10                                 |
     DUMPSRV | __   ECPU%                 6                                 |
     XCFAS   | __   CPUCrit               7                                 |
     GRS     | __   StorCrit              8                                 |
     SMXC    | __   RptClass              8                                 |
     SYSBMAS | __   MemLimit              8                                 |
     SMSVSAM | __   Tran-Act             10                                 |
     CONSOLE | __   Tran-Res             10                                 |
     WLM     |  F1=Help     F2=Split    F3=Cancel   F5=Refresh  F6=Default  |
     ANTMAIN |  F7=Bkwd     F8=Fwd      F9=Swap    F11=Clear   F12=Cancel   |
     ANTAS00
     OMVS     OMVS     OMVS              SC63               NS  FF 7200   0.00
     IEFSCHAS IEFSCHAS                   SC63               NS  FF   73   0.00
     JESXCF   JESXCF   IEFPROC           SC63               NS  FF  540   0.00
     ALLOCAS  ALLOCAS                    SC63               NS  FF  302   0.00
     IOSAS    IOSAS    IEFPROC           SC63               NS  FF  47T   0.00
     IXGLOGR  IXGLOGR  IEFPROC           SC63               NS  FF  13T   0.00
     SMS      SMS      IEFPROC           SC63               NS  FE  429   0.00
```

Figure 5-17 Arrange pop-up to customize MemLimit column

The MemLimit column is shown only for OS/390 R12 systems running in z/Architecture mode. If no systems included on the sysplex DA panel qualify, then the column is suppressed.

The source for the column is R791MEML in the SMF 79 subtype 1 record.

5.5 Logger multiblock exploitation

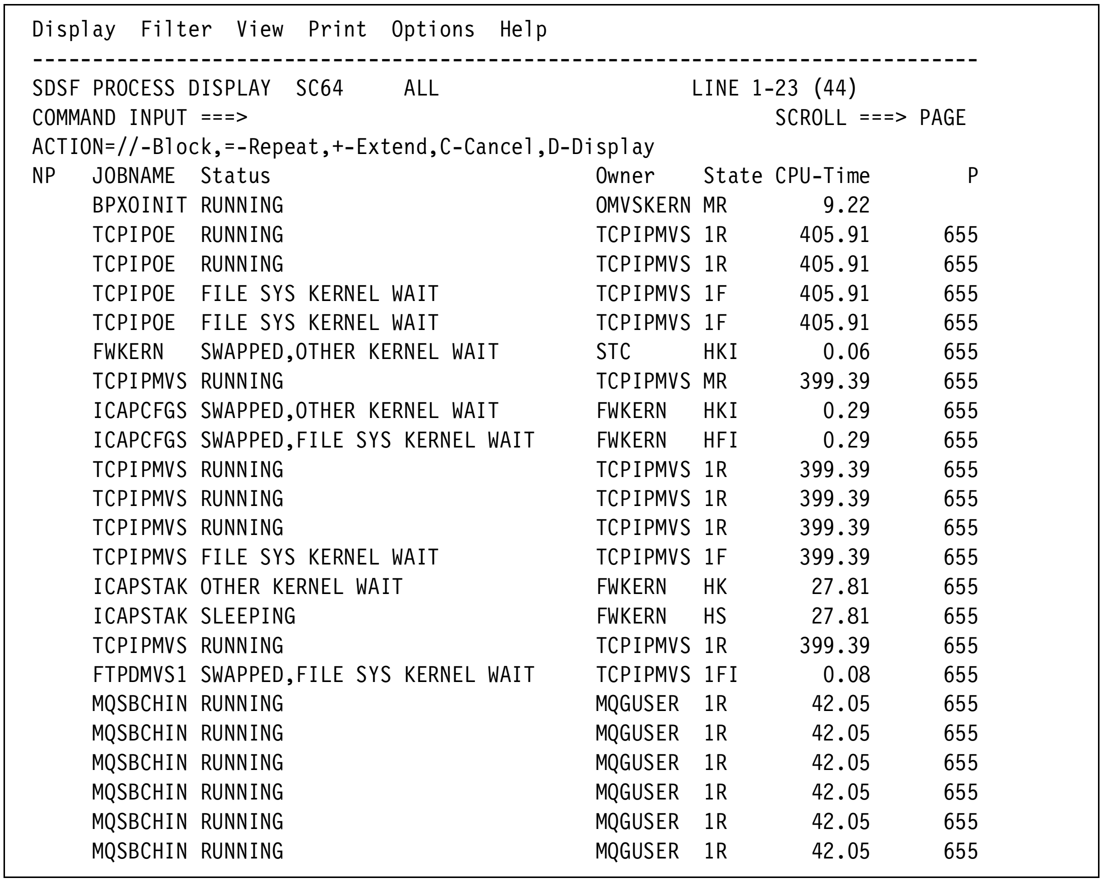
Before OS/390 Release 10, only one log block could be read per IXGBRWSE request at a time, although the logger did some buffering of data read from a log stream. This led to linkage overhead and front-end processing overhead, especially for the Operlog display with SDSF.

With the introduction of the MULTIBLOCK keyword on the IXGBRWSE macro, the need for repetitive calls to the IXGBRWSE service to obtain consecutive log blocks is greatly reduced. This is done by increasing the DASD I/O buffer size used by the logger from approximately one quarter cylinder to half a cylinder.

SDSF in this release exploits the multiblock interface added to System Logger in OS/390 R10. The multiblock interface is intended to improve performance when reading a logstream, and should improve the performance of SDSF's Operlog display.

5.6 z/OS UNIX display

SDSF adds the ability to display z/OS UNIX Systems Services (z/OS UNIX) data for an address space, similar to the **D OMVS** command, which includes processes, parent processes, the state of the process, and the time the process was started. This data is displayed on a new panel by entering **PS** on the command line, and is displayed as shown in Figure 5-18.



```
Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF PROCESS DISPLAY  SC64     ALL                     LINE 1-23 (44)
COMMAND INPUT ===>                                            SCROLL ===> PAGE
ACTION=//-Block,=-Repeat,+-Extend,C-Cancel,D-Display
NP   JOBNAME  Status                     Owner    State CPU-Time        P
     BPXOINIT RUNNING                    OMVSKERN MR         9.22
     TCPIPOE  RUNNING                    TCPIPMVS 1R       405.91      655
     TCPIPOE  RUNNING                    TCPIPMVS 1R       405.91      655
     TCPIPOE  FILE SYS KERNEL WAIT       TCPIPMVS 1F       405.91      655
     TCPIPOE  FILE SYS KERNEL WAIT       TCPIPMVS 1F       405.91      655
     FWKERN   SWAPPED,OTHER KERNEL WAIT  STC      HKI        0.06      655
     TCPIPMVS RUNNING                    TCPIPMVS MR       399.39      655
     ICAPCFGS SWAPPED,OTHER KERNEL WAIT  FWKERN   HKI        0.29      655
     ICAPCFGS SWAPPED,FILE SYS KERNEL WAIT  FWKERN   HFI        0.29      655
     TCPIPMVS RUNNING                    TCPIPMVS 1R       399.39      655
     TCPIPMVS RUNNING                    TCPIPMVS 1R       399.39      655
     TCPIPMVS RUNNING                    TCPIPMVS 1R       399.39      655
     TCPIPMVS FILE SYS KERNEL WAIT       TCPIPMVS 1F       399.39      655
     ICAPSTAK OTHER KERNEL WAIT          FWKERN   HK        27.81      655
     ICAPSTAK SLEEPING                   FWKERN   HS        27.81      655
     TCPIPMVS RUNNING                    TCPIPMVS 1R       399.39      655
     FTPDMVS1 SWAPPED,FILE SYS KERNEL WAIT  TCPIPMVS 1FI        0.08      655
     MQSBCHIN RUNNING                    MQGUSER  1R        42.05      655
     MQSBCHIN RUNNING                    MQGUSER  1R        42.05      655
     MQSBCHIN RUNNING                    MQGUSER  1R        42.05      655
     MQSBCHIN RUNNING                    MQGUSER  1R        42.05      655
     MQSBCHIN RUNNING                    MQGUSER  1R        42.05      655
     MQSBCHIN RUNNING                    MQGUSER  1R        42.05      655
```

Figure 5-18 Processes display

Like other SDSF displays, the new display is sysplex-wide, with SDSF servers and MQSeries for OS/390 V2 gathering and communicating sysplex-wide data. Interaction with the panel is with action characters. There are no overtypeable columns on the PS panel.

The SDSF Primary Option Menu is updated with the PS - Processes as shown Figure 5-1 on page 77.

The Processes choice is also added to the Display pull-down option.

5.6.1 PS commands

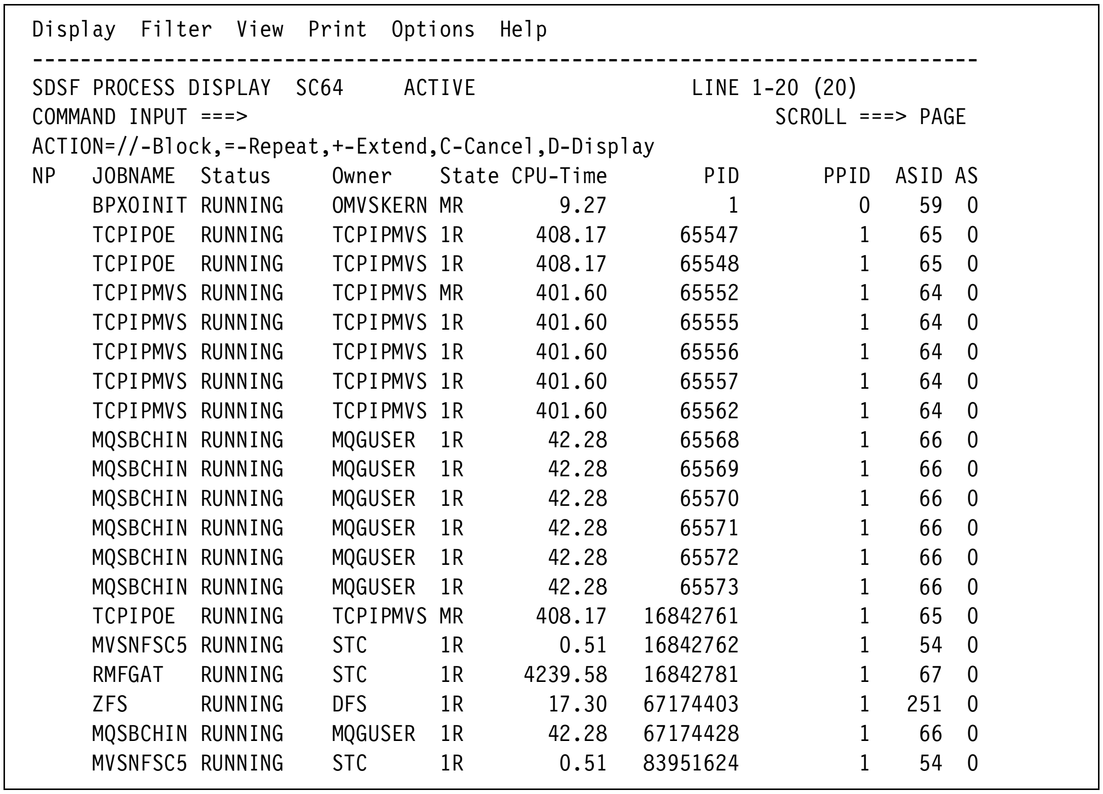
The new **PS** command displays the processes panel. Parameters let you display all processes or only active ones. The syntax is as follows:

PS (ACTIVE | ALL)

ALL Displays all OS/390 UNIX processes. This is the default.

ACTIVE Display only active OS/390 UNIX processes.

Using the command, **ps active**, displayed in Figure 5-19, only the active processes are displayed. The Status column has been modified using the **Arrange** command to change the field width from 32 to 10.



```
Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF PROCESS DISPLAY  SC64     ACTIVE                 LINE 1-20 (20)
COMMAND INPUT ===>                                           SCROLL ===> PAGE
ACTION=//-Block,=-Repeat,+-Extend,C-Cancel,D-Display
NP   JOBNAME  Status     Owner    State CPU-Time        PID       PPID  ASID AS
     BPXOINIT RUNNING    OMVSKERN MR        9.27          1          0    59  0
     TCPIPOE  RUNNING    TCPIPMVS 1R      408.17      65547          1    65  0
     TCPIPOE  RUNNING    TCPIPMVS 1R      408.17      65548          1    65  0
     TCPIPMVS RUNNING    TCPIPMVS MR      401.60      65552          1    64  0
     TCPIPMVS RUNNING    TCPIPMVS 1R      401.60      65555          1    64  0
     TCPIPMVS RUNNING    TCPIPMVS 1R      401.60      65556          1    64  0
     TCPIPMVS RUNNING    TCPIPMVS 1R      401.60      65557          1    64  0
     TCPIPMVS RUNNING    TCPIPMVS 1R      401.60      65562          1    64  0
     MQSBCHIN RUNNING    MQGUSER  1R       42.28      65568          1    66  0
     MQSBCHIN RUNNING    MQGUSER  1R       42.28      65569          1    66  0
     MQSBCHIN RUNNING    MQGUSER  1R       42.28      65570          1    66  0
     MQSBCHIN RUNNING    MQGUSER  1R       42.28      65571          1    66  0
     MQSBCHIN RUNNING    MQGUSER  1R       42.28      65572          1    66  0
     MQSBCHIN RUNNING    MQGUSER  1R       42.28      65573          1    66  0
     TCPIPOE  RUNNING    TCPIPMVS MR      408.17   16842761          1    65  0
     MVSNFSC5 RUNNING    STC      1R        0.51   16842762          1    54  0
     RMFGAT   RUNNING    STC      1R     4239.58   16842781          1    67  0
     ZFS      RUNNING    DFS      1R       17.30   67174403          1   251  0
     MQSBCHIN RUNNING    MQGUSER  1R       42.28   67174428          1    66  0
     MVSNFSC5 RUNNING    STC      1R        0.51   83951624          1    54  0
```

Figure 5-19   Display of the processes using the PS command

Controlling display options

The **PREFIX** and **OWNER** commands, which filter the contents of SDSF panels based on jobname and owner, apply to the new PS panel.

The **QUERY AUTH** command is updated to include PS in the response.

The **SELECT** command is updated to filter PS based on the fixed field of the display (JOBNAME).

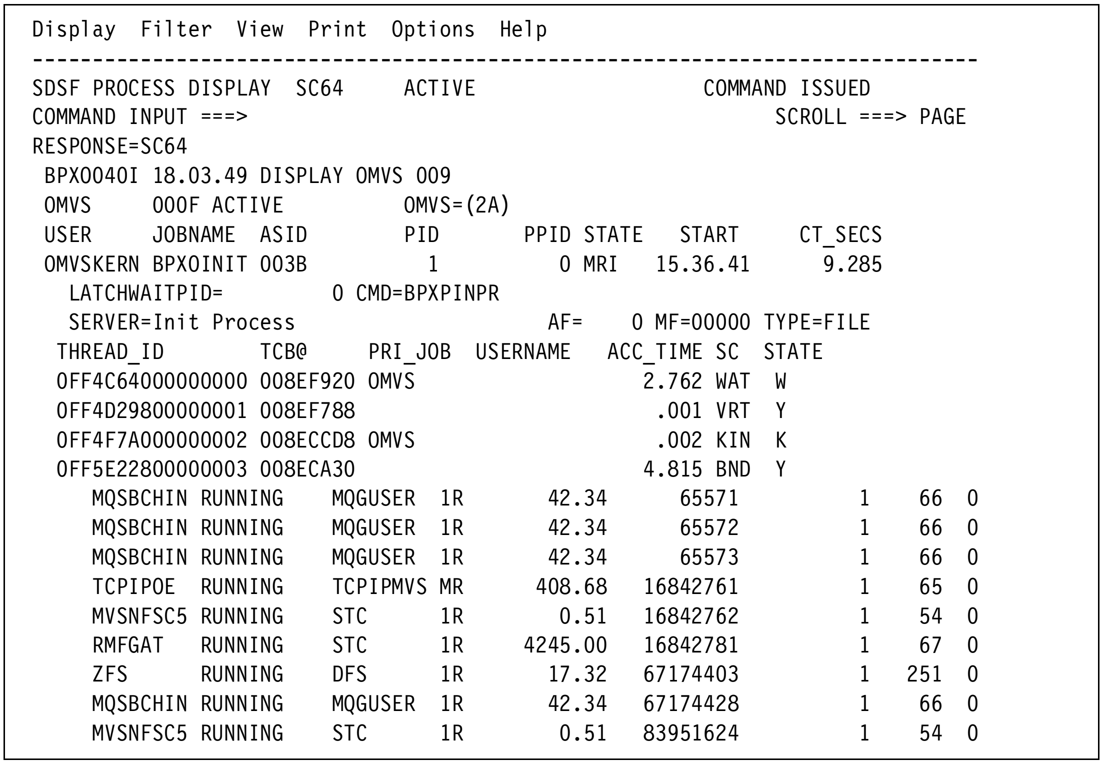
The **SET ACTION** command is updated to display the action characters for the PS display. The action characters and the text for the long form of **SET ACTION** are shown in Table 5-5.

Table 5-5   Action characters for the PS panel

Character	Description	Command	Confirm
C	Cancel the process	C jobname,A= C U=userid C U=*userid*,A=	Yes
D	Display	D OMVS,PID=	No

An example of the D action character, if placed on the BPXOINIT jobname line in Figure 5-19, is displayed in Figure 5-20 on page 94. SDSF issues the following command for the D action character:

```
d omvs,pid=1
```



```
Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF PROCESS DISPLAY  SC64     ACTIVE                   COMMAND ISSUED
COMMAND INPUT ===>                                            SCROLL ===> PAGE
RESPONSE=SC64
 BPXO040I 18.03.49 DISPLAY OMVS 009
 OMVS     000F ACTIVE          OMVS=(2A)
 USER     JOBNAME  ASID        PID       PPID STATE   START     CT_SECS
 OMVSKERN BPXOINIT 003B          1          0 MRI   15.36.41      9.285
   LATCHWAITPID=         0 CMD=BPXPINPR
   SERVER=Init Process                     AF=    0 MF=00000 TYPE=FILE
  THREAD_ID        TCB@     PRI_JOB  USERNAME   ACC_TIME SC  STATE
  0FF4C64000000000 008EF920 OMVS                   2.762 WAT  W
  0FF4D29800000001 008EF788                         .001 VRT  Y
  0FF4F7A000000002 008ECCD8 OMVS                    .002 KIN  K
  0FF5E22800000003 008ECA30                        4.815 BND  Y
     MQSBCHIN RUNNING    MQGUSER  1R       42.34      65571          1    66  0
     MQSBCHIN RUNNING    MQGUSER  1R       42.34      65572          1    66  0
     MQSBCHIN RUNNING    MQGUSER  1R       42.34      65573          1    66  0
     TCPIPOE  RUNNING    TCPIPMVS MR      408.68   16842761          1    65  0
     MVSNFSC5 RUNNING    STC      1R        0.51   16842762          1    54  0
     RMFGAT   RUNNING    STC      1R     4245.00   16842781          1    67  0
     ZFS      RUNNING    DFS      1R       17.32   67174403          1   251  0
     MQSBCHIN RUNNING    MQGUSER  1R       42.34   67174428          1    66  0
     MVSNFSC5 RUNNING    STC      1R        0.51   83951624          1    54  0
```

Figure 5-20 Display of the D action character for jobname BPXOINIT

Updating ISFPARMS for PS panel

The primary and alternate field list for the PS panel in ISFPARMS can be defined using the PSFLDS and PSFLD2 keywords in the ISFGRP macro or the GROUP statement. By default, the two field lists are the same. The columns are described in Table 5-6.

Table 5-6 Columns on the PROCESS (PS) Panel

Column (SDSF name)	Title (displayed)	Width	Description	Access immed	Access delay
(fixed field)	JOBNAME	8	Job name	Yes	
STATUS	Status	32	Status of the process	Yes	
OWNERID	Owner	8	User ID of owner	Yes	
STATE	State	5	State of the process or of most recent created thread	Yes	
CPU	CPU-Time	8	Compute time in hundredths of seconds	Yes	
PID	PID	10	Process ID	Yes	
PPID	PPID	10	Parent process ID	Yes	
ASID	ASID	5	Address space id	Yes	
ASIDX	ASIDX	5	Address space id in hexadecimal	Yes	
LATCHPID	LatchWaitPID	12	PID on which this process is waiting	Yes	
COMMAND	Command	40	Command that created process	Yes	

Column (SDSF name)	Title (displayed)	Width	Description	Access immed	Access delay
SERVER	ServerName	32	Server name	Yes	
TYPE	Type	4	Server type (only when the process is a server)	Yes	
ACTFILES	Actfiles	8	Maximum number of files (only when the process is a server)	Yes	
MAXFILES	MaxFiles	8	Maximum number of files (only when the process is a server)	Yes	
TIMEE	St-Time	8	Time process was started	Yes	
DATEE	St-Date	8	Date process was started	Yes	
SYSLEVEL	Syslevel	25	Level of operating system	Yes	
SYSNAME	SysName	8	System name where process is executing	Yes	

The PREFIX, OWNER, ISTATUS and XSTATUS parameters of ISFPARMS, which control which rows are displayed on SDSF panels based on jobname and owner, apply to the PS panel.

5.7 Filter enhancements

z/OS V1R2 SDSF enhances the filter function to make it easier to use the **`filter`** command. In previous releases, when filters were turned off for a display or when a new filter was created, previously defined filters were discarded. This caused filters to be retyped repeatedly. With the new enhancement, SDSF introduces filters that are defined but not active, and the ability to use the **`filter`** command to add a new filter to already existing filters.

5.7.1 Using filter options

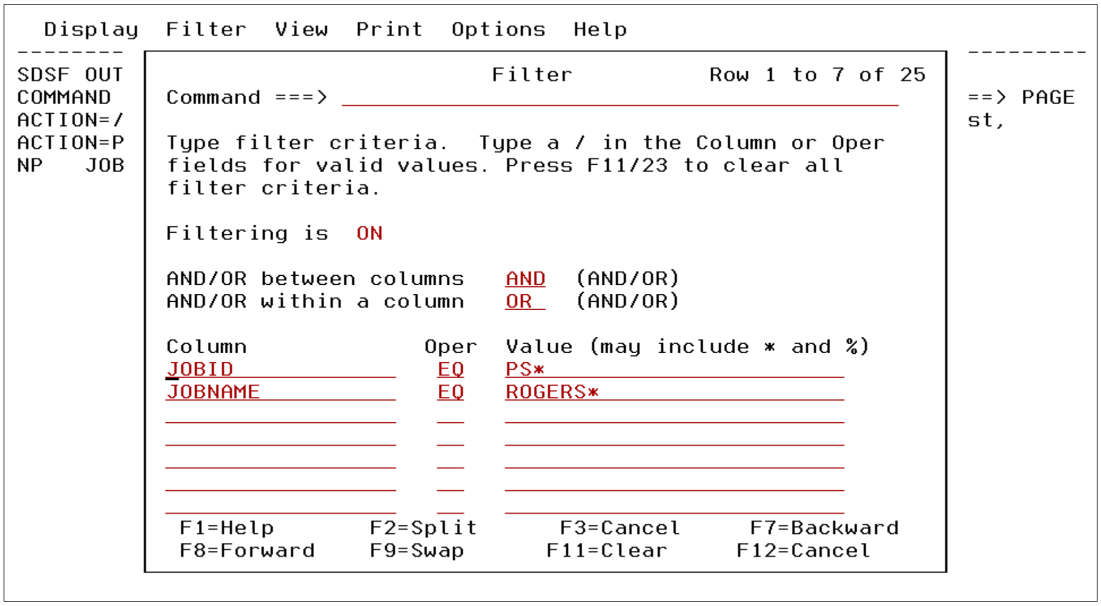
Adding filters with the command is available only when SDSF is running as an ISPF dialog. The function is not extended to the other environments, such as TSO, because without the ISPF dialog services it is difficult to display multiple filters in a usable format. The **`filter`** command and support is enhanced in the following ways:

- It allows you to add a filter to or remove a filter from existing filters.
- The OFF parameter is changed to disable but retain filters; previously, OFF caused filters to be discarded.
- Filters are saved across sessions only under ISPF.
- Filter criteria from previous releases are retained and are compatible with the new support.
- Filters saved with z/OS V1R2 SDSF work with lower levels of SDSF.
- The on/off support is not available with lower levels of SDSF.

To display which filter options are defined for a particular display, on the command line enter:

```
filter ?
```

The response is shown in Figure 5-21 on page 96.



```
  Display  Filter  View  Print  Options  Help
 --------                                                                ---------
 SDSF OUT                       Filter          Row 1 to 7 of 25
 COMMAND     Command ===> _________________________________________      ==> PAGE
 ACTION=/                                                                st,
 ACTION=P    Type filter criteria.  Type a / in the Column or Oper
 NP   JOB    fields for valid values. Press F11/23 to clear all
             filter criteria.

             Filtering is  ON

             AND/OR between columns    AND  (AND/OR)
             AND/OR within a column    OR   (AND/OR)

             Column              Oper   Value (may include * and %)
             JOBID_____________   EQ    PS*_________________________
             JOBNAME___________   EQ    ROGERS*_____________________
             __________________   __    ____________________________
             __________________   __    ____________________________
             __________________   __    ____________________________
             __________________   __    ____________________________
             __________________   __    ____________________________
              F1=Help       F2=Split      F3=Cancel     F7=Backward
              F8=Forward    F9=Swap      F11=Clear     F12=Cancel
```

Figure 5-21 Display current filters defined

When multiple filters are defined for a display, there is an AND or OR relationship among them.

Where:

AND This condition is used if the filters are for different columns

OR This condition is used if the filters are for the same column

5.7.2 DEST command enhancement

For consistency, the DEST command, which already has parameters to allow adding and deleting destinations, is enhanced to accept the new syntax. The command can now be entered with no blank between the + or - and the destination name, for example:

```
dest +poke
dest -pokeps
```

The first command, **`dest +poke`** adds the destination while the second command removes the destination.

5.8 SDSF server enhancements

SDSF makes it easier to use and manage the SDSF server, with these enhancements:

- A default server is provided.
- A new server startup option, LOGTYPE, which controls the destination of the server log, with options for FILE and HARDCPY. Operators can change the option with the server MODIFY command. The **`Display`** command displays the current setting (F server-name D).
- MQ clustering support to simplify definition of queues. The COMM statement in ISFPARMS is enhanced with new keywords.

5.8.1 Default server

The default server eliminates the necessity for the SERVER keyword in the assembler ISFPARMS to identify which server a user connects to. You need only to use the dynamic ISFPARMS.

The default server is identified by a CONNECT statement in ISFPARMS. The CONNECT statement can be placed anywhere in the ISFPARMS statements. There can be only one default server on a system; each system in the sysplex can have a default server. The syntax is as follows:

```
CONNECT DEFAULT(YES | NO | COND)
```

YES Indicates this server is to be made the default server unconditionally, replacing any other default server if necessary.

NO Indicates the server is not to be made the default. If the server had previously been made the default, it remains the default until it terminates.

COND Indicates that the server is to be the default unless another default server is already defined.

That default server processes the dynamic ISFPARMS and is also the local server if communication is used to gather sysplex data. SDSF connects to a server using the server name, which is assigned when the server starts.

With this new support, each SDSF server processes ISFPARMS when it is started. If the ISFPARMS identifies the server as a default server, a flag is set. Users getting into SDSF, then, connect to that default server if no other server is identified through the SDSF command or ISFPARMS.

The SDSF client tries to connect to a server as follows:

1. Server specified on the SDSF command
2. Server specified on the SERVER keyword of the assembler ISFPARMS
3. Default server
4. Server SDSF

5.8.2 Server logging

The new logging option, **LOGTYPE**, is added as a parameter on the **START** and **MODIFY** commands. The logging option for these commands is as follows:

LOGTYPE or **LT** Specifies the destination of the server log. The options are as follows:

FILE or **F** Specifies that the report will be written to file with the ddname SDSFLOG. This is the default, unless the SDSF server is running under MSTR.

HARDCPY or **H** Specifies that messages issued during processing of ISFPARMS will be written to the hardcopy log (syslog). This is the default if the SDSF server is running under MSTR.

The summary report includes each error message that is issued. This saves time in locating error messages, which can be difficult to find in the server log. Since most of the error messages include the statement and line in the text, the report is all that is needed to quickly find an error.

Server start command

Use the server **START** command to initialize the SDSF server address space and to control server options. When the server is initialized, the server is ready to process requests from the SDSF application. You can use any of the following start options:

- **S SDSF**

  This command starts the SDSF server address space, with the name SDSF.

- **S SDSF,M=01**

  This command starts the SDSF server address space, with the name SDSF. Statements will be read from member ISFPRM01 of the data set defined in the server JCL. Member ISFPRM01 is made the default member for any subsequent MODIFY server, REFRESH commands.

- **S SDSF,M=01,P='FM,LC(H)'**

  This command starts the SDSF server address space, with the name SDSF. Statements will be read from member ISFPRM01 of the data set defined in the server JCL. Server messages will be folded to uppercase. The default SYSOUT class for the server log is H.

- **S SDSF,P='LT(HARDCPY)'**

  This command starts the SDSF server address space with the name SDSF. The server log will be written to the hardcopy log.

- **S SDSFT**

  This command starts the SDSF server address space with the name SDSFT.

- **S SDSF.SDSFA**

  This command starts the SDSF server address space with proc SDSF and server name SDSFA. The server name, SDSFA, corresponds to the name coded in the ISFPMAC macro of ISFPARMS, or on the server command.

Note: When tracing is active, significant performance degradation may occur. A significant amount of trace output may be generated.

Server modify command

Use the **MODIFY** command to dynamically change server options. You can specify a test mode to cause the syntax of the statements to be checked without activating the statements. You can use any of the following modify options:

- **F SDSF,LC(H)**

  This command changes the default SYSOUT class for the server log to H.

Use the **MODIFY,D** command to display options for the SDSF server.

- **F SDSF,D,TRACE**

  This command displays the current setting for trace.

5.9 Minor SDSF enhancements

SDSF adds several minor enhancements.

5.9.1 User symbols on WHEN statement

User symbols are supported on the WHEN statement in ISFPARMS. Previously, the WHEN statement was limited to constants, such as system name, server name, and userID. With the new support, it is possible to have a WHEN check for a specified value on any system static symbol.

The format is WHEN SYMBOL(x=|^=y,...) where x and y can be strings or symbols. A symbol is expressed as &name. Multiple conditions can be specified, separated by a comma. For the “not equal” condition, ^=, /= and \= are all accepted.

There is no limit to the number of conditions that can be specified; all must be true for the statement to be accepted. The SYMBOL keyword can be combined with any other WHEN keyword, and all keywords must evaluate to true to be accepted. If duplicate SYMBOL keywords are present, the last one replaces any prior ones, regardless of the previous conditions that were processed (i.e. conditions cannot be replaced individually). For the “equal” condition, the strings must match in length and content. Strings are case sensitive. The operands can be specified in either order (&SYSNAME=SYS1 or SYS1=&SYSNAME). If an operand does not evaluate to a symbol, the string is checked as is.

Some examples of the use of the WHEN statement are:

- WHEN SYMBOL (&SYSNAME ^= SY1)

  This is accepted when the value of symbol SYSNAME is not equal to SY1. Note that this will also be accepted if SYSNAME is not a defined symbol, as the character string &SYSNAME is not equal to the string SY1.

- WHEN SYMBOL (SY1 ^= &SYSNAME)

  Like the previous example, this is accepted when the value of symbol SYSNAME is not equal to SY1.

- WHEN SYMBOL (&SYSNAME = SY1, &SYSPLEX = PLEX1)

  This is accepted when the value of symbol SYSNAME is equal to SY1, and the value of symbol SYSPLEX is equal to PLEX1.

- WHEN SYMBOL (&SYSPLEX = PLEX1) SYSNAME(SY1)

  This example shows a WHEN with two conditions, one of which uses a symbol. This WHEN is accepted when the value of the symbol SYSPLEX is PLEX1 and the sysname is SY1.

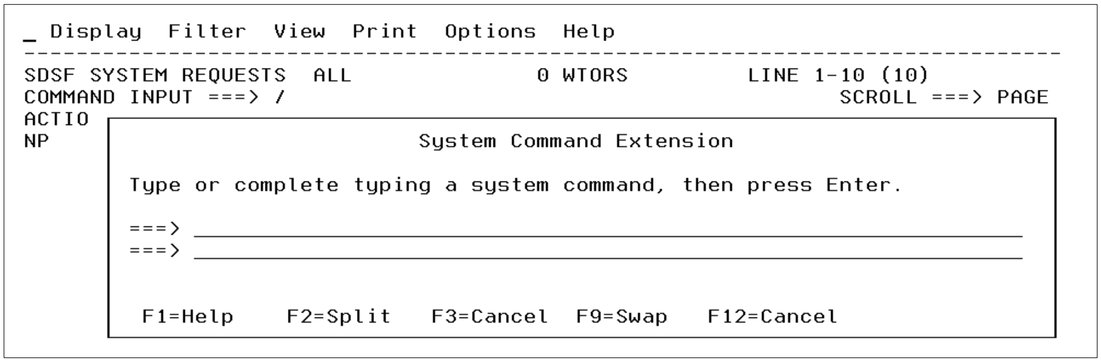
- WHEN SYMBOL (&KEN = &KEN)

  This is always true. If KEN is not a defined symbol, then &KEN is treated as a character string.

5.9.2 System Command Extension pop-up enhancements

The System Command Extension pop-up, for long / commands, is updated to make it less likely that commands will break across input lines. When a command breaks across input lines, users cannot use input mode to modify it. The update is to change the ISPF pop-up to use 2 input lines rather than 3, as shown in Figure 5-22 on page 100. The TSO version of the pop-up is not changed.

Because it never displays previous commands, the Reply pop-up, which is used to enter replies on the SR panel, is not changed.



```
_ Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF SYSTEM REQUESTS  ALL              0 WTORS         LINE 1-10 (10)
COMMAND INPUT ===> /                                         SCROLL ===> PAGE
ACTIO
NP                         System Command Extension

        Type or complete typing a system command, then press Enter.

        ===> ________________________________________________________________
        ===> ________________________________________________________________


         F1=Help    F2=Split   F3=Cancel  F9=Swap   F12=Cancel
```

Figure 5-22   System Command Extension pop-up

5.9.3 New elapsed transaction time columns on DA

SDSF adds two new columns to show elapsed transaction time to the end of the field lists on the DA panel, as shown in Table 5-7.

Table 5-7   Elapsed transaction time columns on DA panel

Column (SDSF name)	**Title (displayed)**	**Width**	**Description**	**Access immed**	**Access delay**
TRANACT	Tran-Act	10	Elapsed time the transaction has been active	Yes	
TRANRES	Tran-Res	10	Elapsed time the transaction was swapped in	Yes	

The columns are in hhhh:mm:ss.th format. With the default width, the digits to the right of the decimal point are not visible. Users can increase the column width to see the full value.

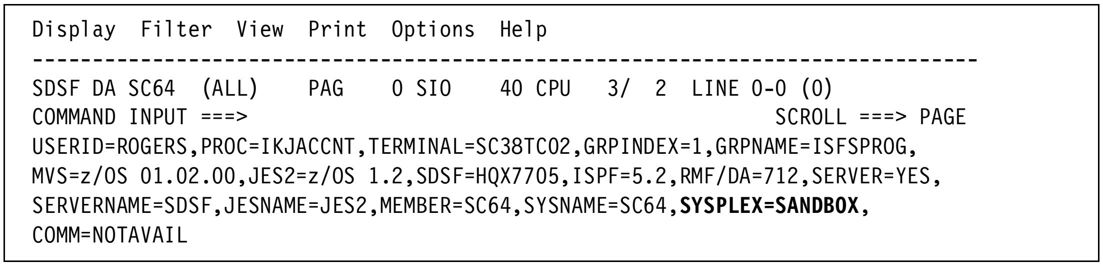
The Filter function is changed to accept times in the hhhhh:mm:ss.th format. As before, leading zeros are optional for the hours. For example, a value for the St-Time field can be specified as 23:01:01, 023:01:01, or 0023:01:01. However, 40:01:01 is invalid for that field because it is not a valid time of day. For Tran-Act, 40:01:01, 140:01:01, or 1140:01:01 would be valid.

Installations can change the titles, widths and locations of the column with the field lists in ISFPARMS. Users can change the widths and locations of the columns with the Arrange function.

The values for the columns are only collected if the job is swapped in. As a result, the values will be zeros if the job is swapped out.

5.9.4 WHO enhancement

The sysplex name is added to the WHO command, as shown in Figure 5-23 on page 101.



```
Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF DA SC64  (ALL)    PAG    0 SIO    40 CPU   3/  2  LINE 0-0 (0)
COMMAND INPUT ===>                                          SCROLL ===> PAGE
USERID=ROGERS,PROC=IKJACCNT,TERMINAL=SC38TC02,GRPINDEX=1,GRPNAME=ISFSPROG,
MVS=z/OS 01.02.00,JES2=z/OS 1.2,SDSF=HQX7705,ISPF=5.2,RMF/DA=712,SERVER=YES,
SERVERNAME=SDSF,JESNAME=JES2,MEMBER=SC64,SYSNAME=SC64,SYSPLEX=SANDBOX,
COMM=NOTAVAIL
```

Figure 5-23 WHO command display of sysplex name

5.10 Migration and coexistence

As with any new SDSF release, customers must reassemble ISFPARMS with the SDSF macro library. If they have written any user exit routines, they must also assemble ISFUSER.

An SDSF R10 and R12 server can coexist in the same sysplex to provide data for the sysplex displays. (The server must be at the R10 level or higher to provide the sysplex support.) The SDSF client and its local server must be at the same level. For example, a Release 12 client can only connect to a Release 12 server, and a Release 10 client can only connect to a Release 10 server. However, a Release 10 client may request sysplex data from a Release 12 system and a Release 12 client may request data from a Release 10 system. The target system will respond only with the data (columns and panels) supported on that system.

Note: Once the **`$ACTIVATE`** command is issued in the V1R2 environment, JES2 requires that all JES2s in the MAS be at V1R2, and it is no longer possible to communicate with an OS/390 V2R10 SDSF server.

A toleration PTF needs to be applied to SDSF R10 to allow it to correctly process data received from an SDSF R12 server. The PTF will be released concurrently with SDSF R12, and can be applied at any time prior to adding SDSF R12 to a server group that contains an R10 server.

5.10.1 HASPINDX data set

SDSF uses the HASPINDX data set to manage the syslog data sets for use by the SYSLOG panel. The performance of the panel is affected by the activity on the HASPINDX data set. You may share the HASPINDX data set between SDSF users within a JES2 MAS, provided that the SDSF levels are compatible.

For z/OS SDSF, compatible levels are OS/390 V2R5 and higher. Take care not to share the HASPINDX across incompatible environments, as this will cause SDSF to format the data set each time it is accessed. If you have high log activity and many SDSF LOG users, you should consider allocating separate HASPINDX data sets.

There is no limit to the number of HASPINDX data sets you can create, but a reasonable approach would be to create one per system. A HASPINDX can also be allocated to an individual SDSF user for private use. This is done by allocating a data set to ddname (HASPINDX) in the TSO session prior to invoking the **`LOG`** command. In this case, there will be no contention by SDSF users for the HASPINDX, but there is the additional overhead of maintaining the HASPINDX for an individual user.

You define which HASPINDX data set is used in the ISFPMAC macro or OPTIONS statement of ISFPARMS. If you use the statement format ISFPARMS, you can use system symbols to further simplify the definition.

5.10.2 End user commands and syntax

Complete details for all the end user commands and syntax are available from the on-line help.

6

Distributed File Services

In this chapter we discuss the enhancements introduced in z/OS V1R2 Distributed File Services (DFS), and introduce the new z/OS DFS zSeries File System (zFS). The following topics are covered:

- Distributed File Services overview
- The additional RACF definitions and BPXPRMxx entry required to define and run zFS
- How to define zFS as a UNIX System Services physical file system
- How to define the zFS parameter file
- How to start and run zFS in a colony address space
- How to allocate and format a zFS aggregate
- How to create a zFS file system in an aggregate
- How to mount a zFS file system in the USS directory hierarchy
- How to access a zFS file system in the same way as an HFS file system is accessed
- How to run zFS salvage program to validity check or recover an aggregate
- Commands/utilities known to work with zFS
- Known zFS restrictions

6.1 DCE DFS overview

z/OS Distributed File Service (DFS) is an exclusive base element of z/OS Version 1 Release 2. It is a distributed client/server application that presents file systems on different heterogeneous computers in a network as a single global file system. The element includes distributed file service support for the Distributed Computing Environment (DCE), Server Message Block (SMB), file/print server support, and zSeries File System (zFS) support.

The Distributed File Service element is based on the Open Software Foundation (OSF) of the Open Group DCE Distributed File System component at OSF level 1.2.2. DCE Distributed File Services is a network file system that allows you to see and access files on remote computers in the network as if they were in your local file system on your computer. See Figure 6-1 for a graphical overview of DFS.

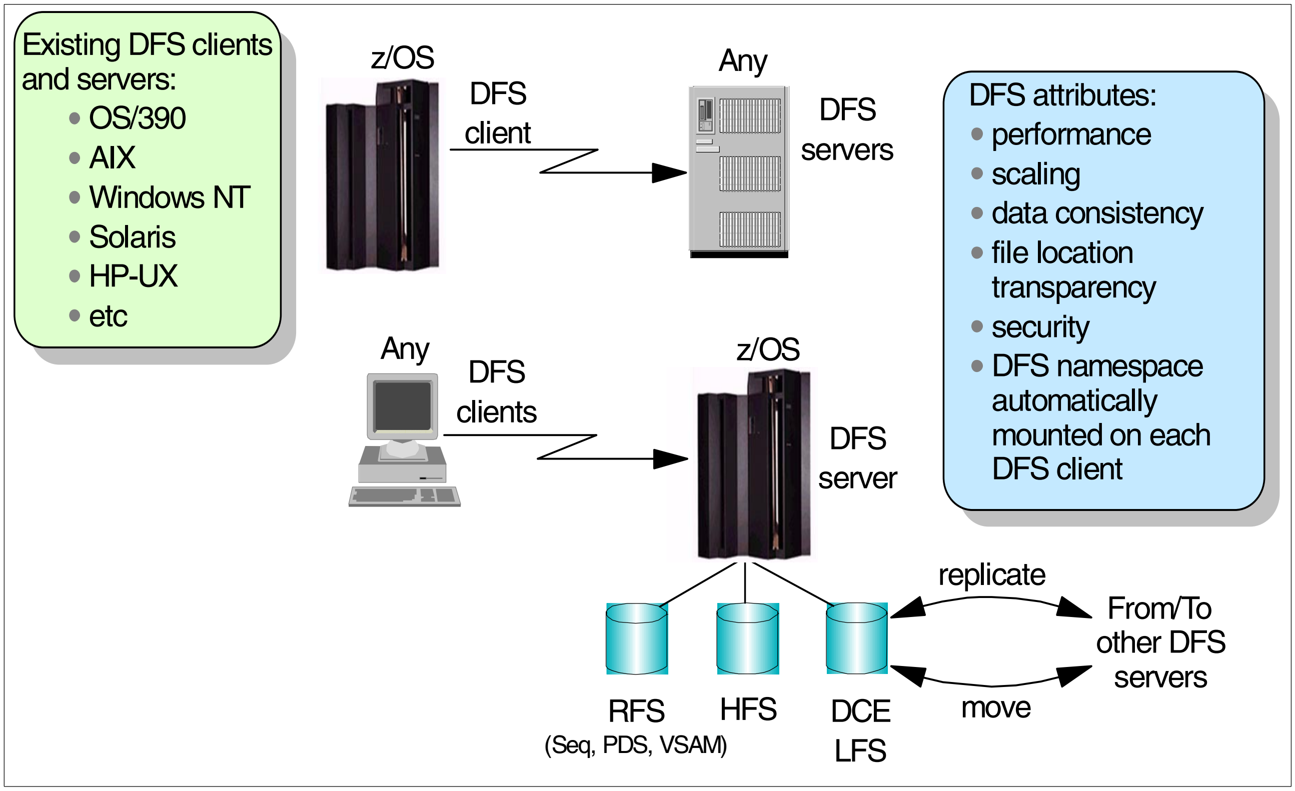
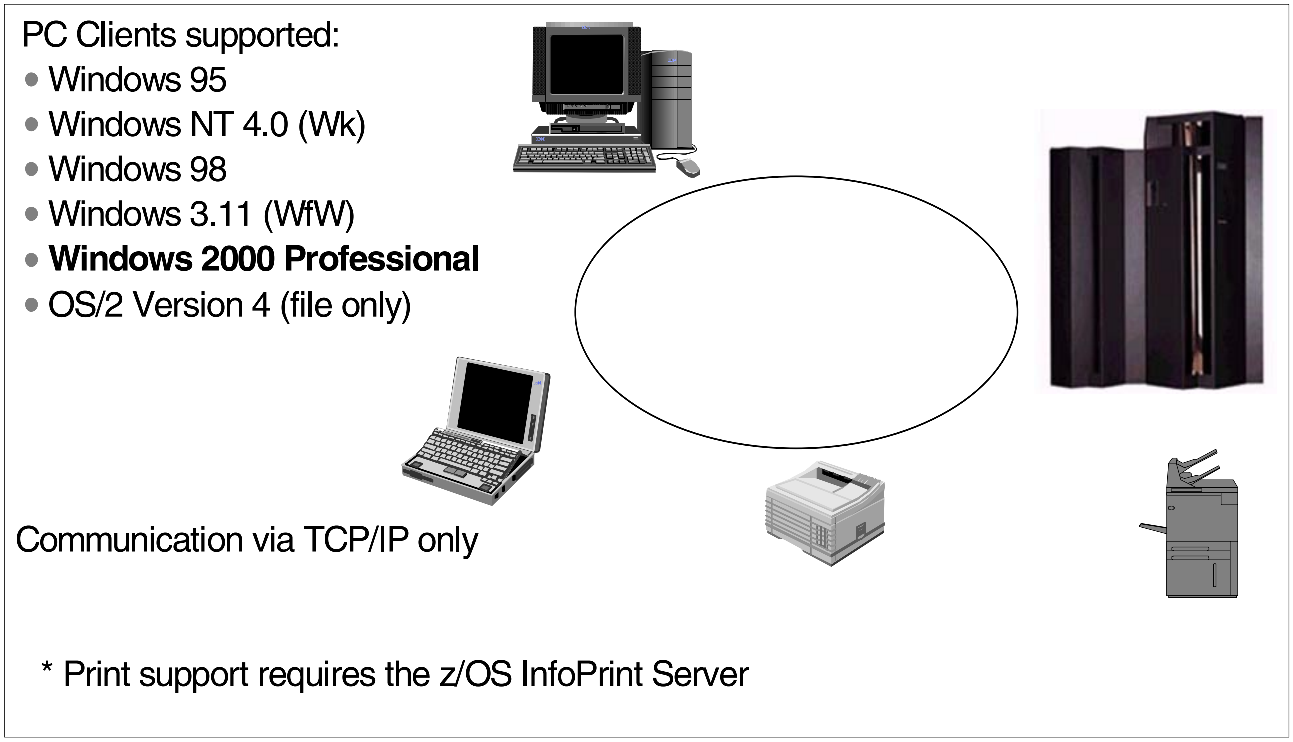


Figure 6-1 DCE DFS overview

6.2 z/OS DFS SMB server

The z/OS Distributed File Service Server Message Block (DFS SMB) file/print server support, or simply z/OS SMB Server, provides a server that makes HFS files and MVS data sets, also referred to as a Record File System (RFS), available to SMB clients. See Figure 6-2 on page 105 for a graphical overview. The SMB protocol is supported only through TCP/IP on z/OS; it allows clients to access shared directory paths and shared printers. PC clients on the network can use file and print sharing as included in the clients operating system, no special software is required. The supported SMB clients include Windows 2000 Professional, Windows 98, and Windows 95.

DCE must be installed on the z/OS SMB server side, but you do not have to configure it or activate it to use the SMB server. z/OS DCE Base Services is an exclusive base element of z/OS V1R2. If you plan to use print serving, the z/OS Infoprint Server, which is an exclusive optional feature of z/OS V1R2, must also be installed on your system. See *Infoprint Server for z/OS Implementation*, SG24-6234.



PC Clients supported:

- Windows 95
- Windows NT 4.0 (Wk)
- Windows 98
- Windows 3.11 (WfW)
- **Windows 2000 Professional**
- OS/2 Version 4 (file only)

Communication via TCP/IP only

* Print support requires the z/OS InfoPrint Server

Figure 6-2 z/OS SMB server

Figure 6-3 on page 106 shows a detailed view of the DFS kernel address space. In z/OS Version 1 Release 2, the zSeries File System is the name for the new UNIX System Services file system, which is based on the DCE LFS (Episode) file system, also shown in Figure 6-3.

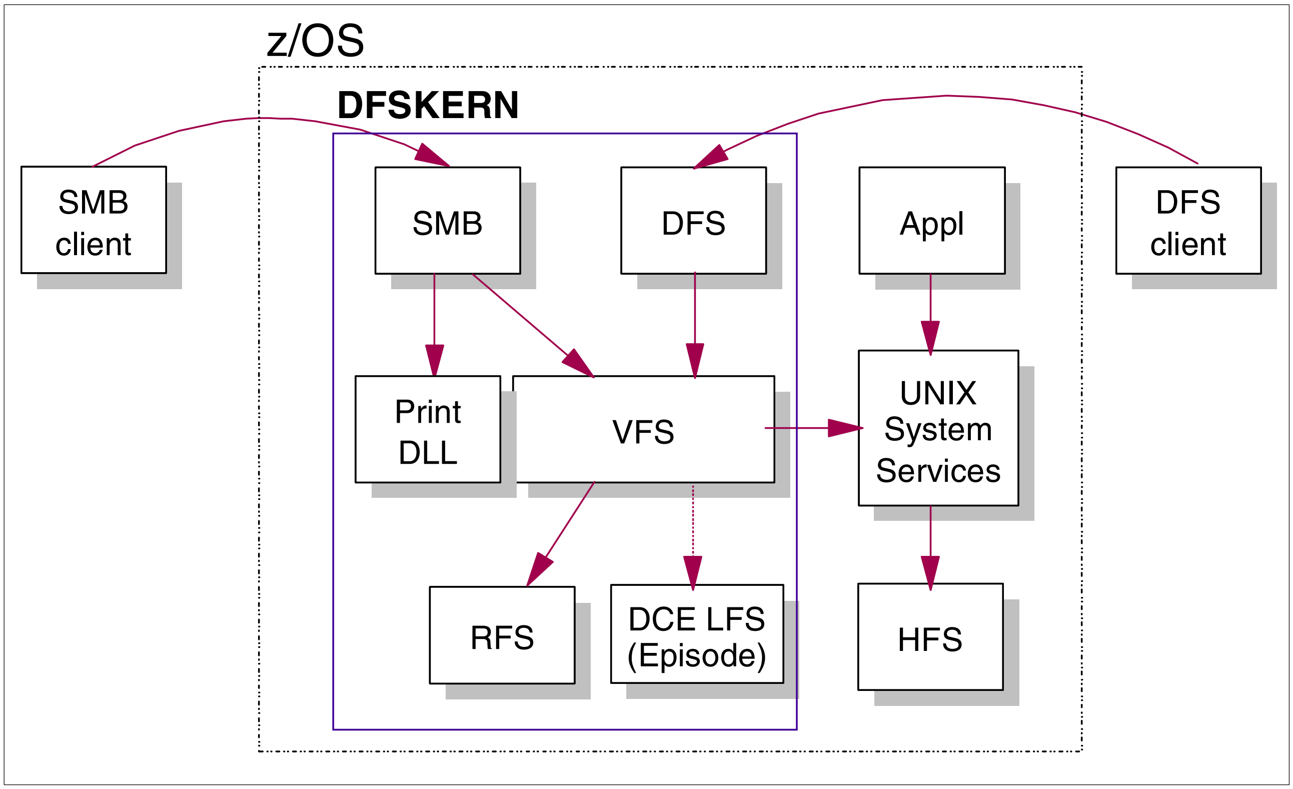


Figure 6-3 DFSKERN, the DFS server

6.3 Distributed File Services

Distributed File Services (DFS) is implemented on most UNIX systems. It is also implemented on Windows NT, and on OS/390 and z/OS. DFS clients can talk to any DFS server implementation, and z/OS DFS server can serve any DFS client implementation. See Figure 6-1 on page 104 for an overview of DFS.

DFS is a base element of z/OS that provides DFS and SMB support. This support allows users to access and share data in a distributed environment across a wide range of computers, including IBM and other platforms. The base element includes client and server support for the DCE.

In you want to use z/OS DFS support, the DCE base element of z/OS must be installed, configured, and running on your system.

z/OS DFS server supports or serves three kinds of file representations:

- Record File System (RFS), which is MVS data sets
- Hierarchical File System (HFS) files
- DCE Local File System (LFS) files

In z/OS V1R2 there are some general serviceability and performance enhancement items that are implemented by the element. These enhancements are internal to the element, and have no external definitions or any installation requirements.

6.4 DFS/SMB support

The z/OS DFS Server Message Block (SMB) support includes file and print serving for Windows clients.

The enhancements in this release include:

- Support for Windows 2000 clients
- Automount
- Enhanced ASCII
- SMB pass-through authentication

6.4.1 Server Message Block

Windows 2000 Professional support is provided. The support is at the same level as the current Windows NT support. With Windows 2000 clients, you should ensure that your computer name (specified in the _IOE_SMB_COMPUTER_NAME environment variable in the /opt/dfslocal/home/dfskern/envar) file is the same as your TCP/IP hostname.

6.4.2 Automount facility

Dynamic export is the basis for the SMB server's support of automounted file systems.

Previously, the SMB server could not export a file system (Shared HFS) that was owned by another system. Now, the SMB server will recognize that it could not export the file system because it was owned by another system and will attempt to "move" the file system to this system where the SMB server that is trying to export it is running.

6.4.3 Enhanced ASCII

The SMB server provides the ability to honor the tag of a file and do the correct data translation. Enhanced ASCII support and file tagging is described in "File tagging overview" on page 132. The SMB server provides character data translation support so it can store text files in EBCDIC but still present the file to a PC user as ASCII. The SMB server can tag a file when the file is created.

There are new environment variables in the dfskern process for this support. These must be enabled in order to use the support.

6.4.4 SMB pass-through authentication

Previously, in order to use the SMB server, you usually needed to synchronize your Windows password with your z/OS password. With this new support, the administrator can configure the SMB server to go to a Windows Domain controller to authenticate the PC user. The SMB client machine must be in the domain of the Domain controller, or be able to be authenticated by that Domain controller.

6.5 zSeries File System

The zSeries File System (zFS) is a "new" UNIX file system that can be used in addition to the Hierarchical File System. The word new is used in quotes since this file system actually has been available since 1995 as part of the DCE component of MVS/ESA V5R2.2, OS/390, and z/OS V1R1. The file system has been known as the DCE DFS Local File System (LFS), sometimes referred to as Episode. It is a high performance, log-based file system. The DCE DSF Local File System is part of the OSF DFS product, and as such, it is included in the Distributed File Service base element of z/OS V1R2. zFS is a separate component of the DFS base element, so it can be serviced separately from the other components of DFS.

zFS introduces some new terms and file system structures that you need to be aware of, specifically:

- The zFS file system
- zFS file system aggregates

  If you are familiar with DFS in a DCE environment, a zFS aggregate is conceptually the same as an Episode aggregate.
- zFS functions
- zFS as a physical file system
- zFS in a colony address space
- zFS installation considerations

The following sections describe the new terminology and customization necessary to run zFS.

6.5.1 zFS file system

zFS focuses on local access. It can be locally mounted and is fully accessible from local UNIX System Services applications. It is also available remotely via the z/OS DFS SMB server from Windows clients.

zFS does not replace HFS, rather it is complementary to HFS. HFS is required for z/OS installation and the root file system must be HFS.

Like HFS, zFS is a UNIX file system. It contains files and directories that can be accessed with the APIs available for HFS. zFS can be mounted into the z/OS UNIX System Services file hierarchy along with other local or remote file system types, such as:

- HFS
- Automount File System (AUTOMNT)
- Temporary File System (TFS)
- Network File System (NFS)

In general, the application view of zFS is the same as the application view of HFS. Once a zFS file system is mounted, it is almost indistinguishable from any other mounted HFS.

The benefits of using zFS are:

- Improved performance
- Underlying architecture to support additional functions
- Improved crash recovery

Initial studies indicate that there will be many environments where zFS will perform better than HFS. Initial test runs show promising results. There are many functions available in DSF LFS that will be made available with zFS in the local access environment.

Metadata cache

The zFS file system has a cache for file system metadata, which includes directory contents and the data of small files (smaller than the aggregate block size). The setting of this cache size is important to performance because zFS references the file system metadata frequently. Synchronous reads of metadata increase I/O rates to disk and server response times.

6.5.2 zFS file system aggregates

zFS supports the use of *aggregates*. A zFS aggregate is a data set that contains zFS file systems. The aggregate is a VSAM Linear Data Set (VSAM LDS) and is a container that can contain one or more zFS file systems. An aggregate can only have one VSAM LDS, but it can contain an unlimited number of file systems. The name of the aggregate is the same as the VSAM LDS name.

A zFS file system is a named entity that resides in a zFS aggregate. It contains a root directory and can be mounted into the UNIX System Services hierarchy. While the term “file system” is not a new term, a zFS file system resides in a zFS aggregate, which is different than an HFS file system.

zFS aggregates comes in two types:

- HFS compatibility mode aggregates

  This type of aggregate can contain only one zFS file system.

- Multi-file system aggregates

  This type of aggregate can contain one or more zFS file systems.

The maximum size of each filesystem in an aggregate is determined when the file system is created. This maximum size number is called a quota.

> **Note:** The term zFS aggregate should not be confused with DFSMS aggregate groups, which is a group of data sets.

Compatibility mode aggregates

A zFS aggregate that contains exactly one single zFS file system is called a compatibility mode aggregate.This is flagged in the aggregate when it is created. The name of the file system is the same as the name of the aggregate, which is the same as the VSAM LDS cluster name. The file system quota in a compatibility mode aggregate is set to the size of the aggregate. Compatibility mode aggregates are more like an HFS data set, except they are VSAM linear data sets instead of HFS data sets. We recommend that you start using compatibility mode aggregates first, since they are more like the familiar HFS data sets. Figure 6-4 on page 110 shows a compatibility mode aggregate.

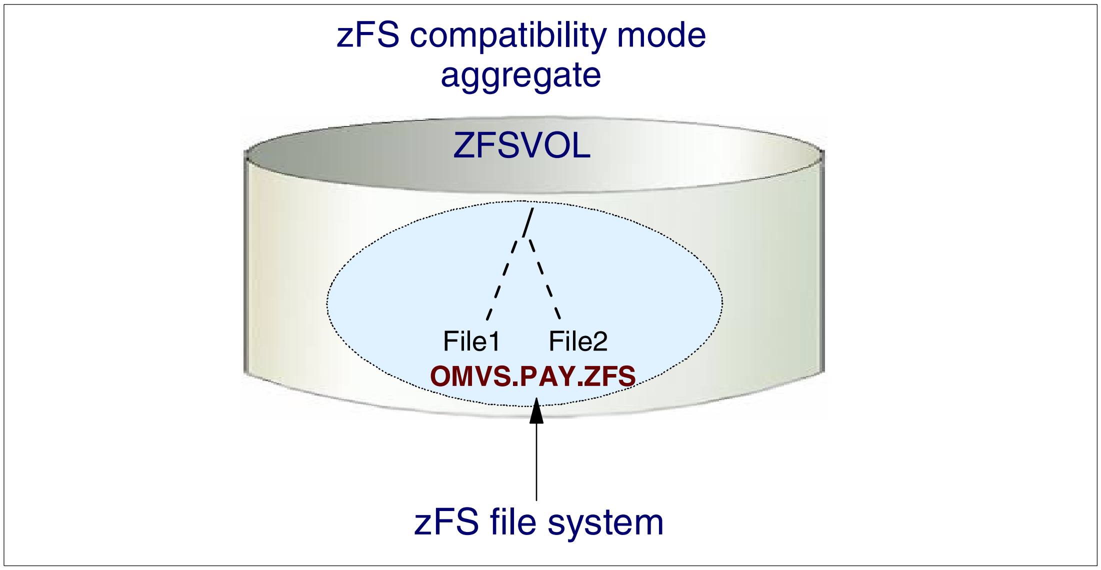


Figure 6-4 Compatibility mode aggregate

Multiple file system aggregates

The multiple file system aggregate OMVS.EBIS.ZFS, shown in Figure 6-5, can contain multiple zFS file systems. This makes it possible to do space sharing between the zFS file systems within the aggregate.

The multiple file system aggregate has its own name. This name is assigned when the aggregate is created. It is always the same as the VSAM LDS cluster name. Each zFS file system in the aggregate has its own file system name. This name is assigned when the particular file system in the aggregate is created. Each zFS file system also has a predefined maximum size, the quota.

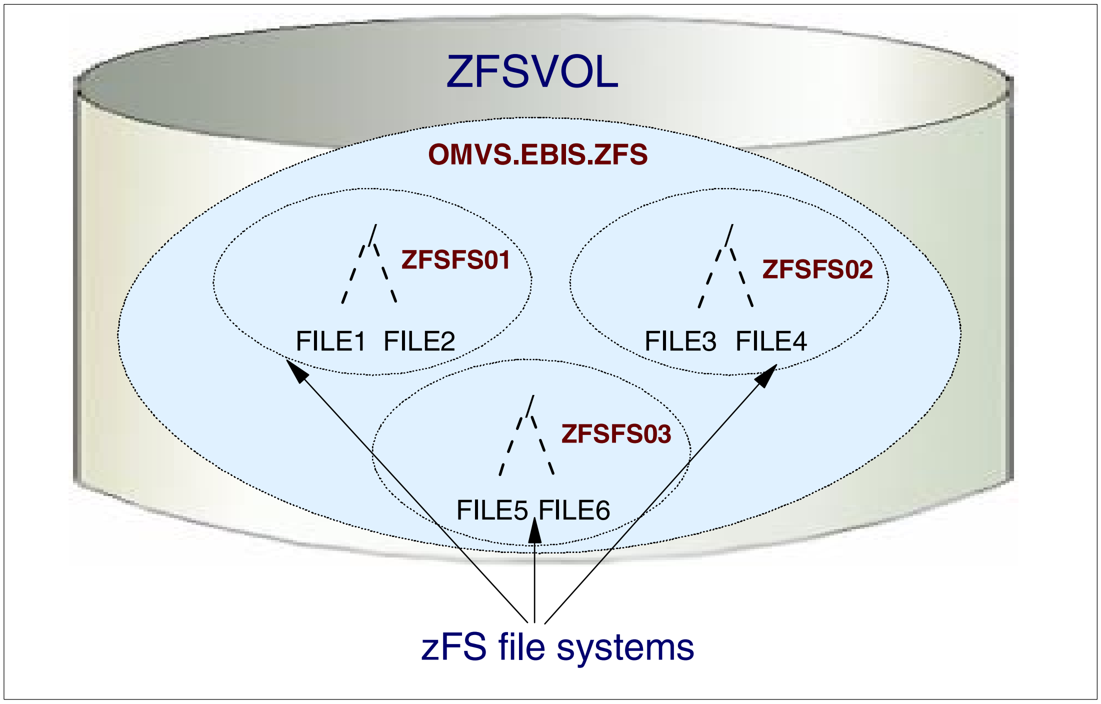


Figure 6-5 Multi-file system aggregate

Aggregate attach

zFS aggregates must be attached by zFS before they can be used. You can think of this as analogous to mounting a file system. They are attached (opened) when the zFS physical file system is started, or they can be attached by a command, **`zfsadm`**.

Since the file system is in the aggregate, the aggregate must be attached before the file system can be mounted. Attach occurs in one of three ways:

1. At IPL time, or when zFS is started, by putting the aggregate name in the zFS parameter file, IOEFSPRM.
2. By issuing the **`zfsadm`** attach command.
3. On the mount of the file system if the aggregate is a compatibility mode aggregate.

A compatibility mode aggregate is more like an HFS and does not require an attach as a separate step.

6.5.3 zFS functions

zFS provides the following new functions to support the use of aggregates:

- Space sharing between file systems in the same data set
- File system maximum size as a logical value
- File system cloning (making a read/only copy)

Space sharing between file systems

Space sharing means that if you have multiple file systems in a single data set, when files are removed from one of the file systems, which frees DASD space, another file system can use that space when new files are created. This new type of file system is called a multi-file system aggregate.

File system size

Since you can now have multiple file systems in a single data set, the maximum file system size becomes a logical limit and is separated from the physical size of the data set. Normally, the file system size would be smaller than the physical size of the data set.

File system cloning

File system cloning is the ability to make a relatively quick read-only copy of a zFS file system that resides in the same aggregate as the original file system, as shown in Figure 6-6 on page 112. It is relatively quick because it does not copy the data blocks: it only copies the metadata. Metadata is file information like the owner and permission bit settings. This means that it is quick and does not take up too much space. The metadata in the clone points to the same data blocks as the metadata in the original file system. The clone file system is given the same name as the original except that **.bak** is appended to the name.

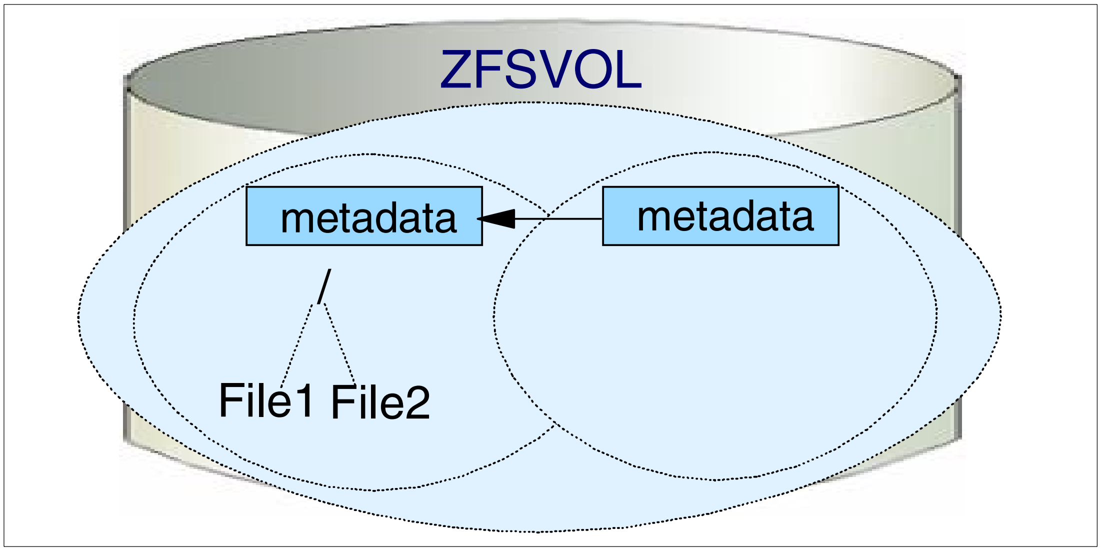


Figure 6-6 zFS file system clone

You need to limit your file system name to 40 characters for any file system that will be cloned. The clone can be mounted and can be used as a complete point-in-time read-only copy of the original file system. If a clone is done every night, every morning users have a read-only copy of yesterday's files. If a user accidently erases a file, the user can go to the read-only clone and copy back yesterday's file with no operator intervention.

As soon as the clone operation is complete, the original file system is immediately available for update. The clone file system can never be updated. When the original file system is updated, new blocks are allocated for the updates, but the original blocks also remain allocated and are still pointed to by the clone metadata. When another clone is done, any original blocks that have been updated in the original file system can be deallocated and made available to the aggregate. Recloning a file system makes more space available in the aggregate.

6.5.4 zFS physical file system

zFS is a physical file system (PFS) that is started by UNIX System Services during the IPL. A physical file system is the part of the operating system that handles the actual storage and manipulation of data on a storage medium.

The PFS interface is a set of protocols and calling interfaces between the logical file system (LFS) and the PFSs that are installed on z/OS UNIX, as shown in Figure 6-7 on page 113. In a UNIX System Services environment, UNIX programs and UNIX users access their files through these interfaces. PFSs mount and unmount file systems and perform other file operations.

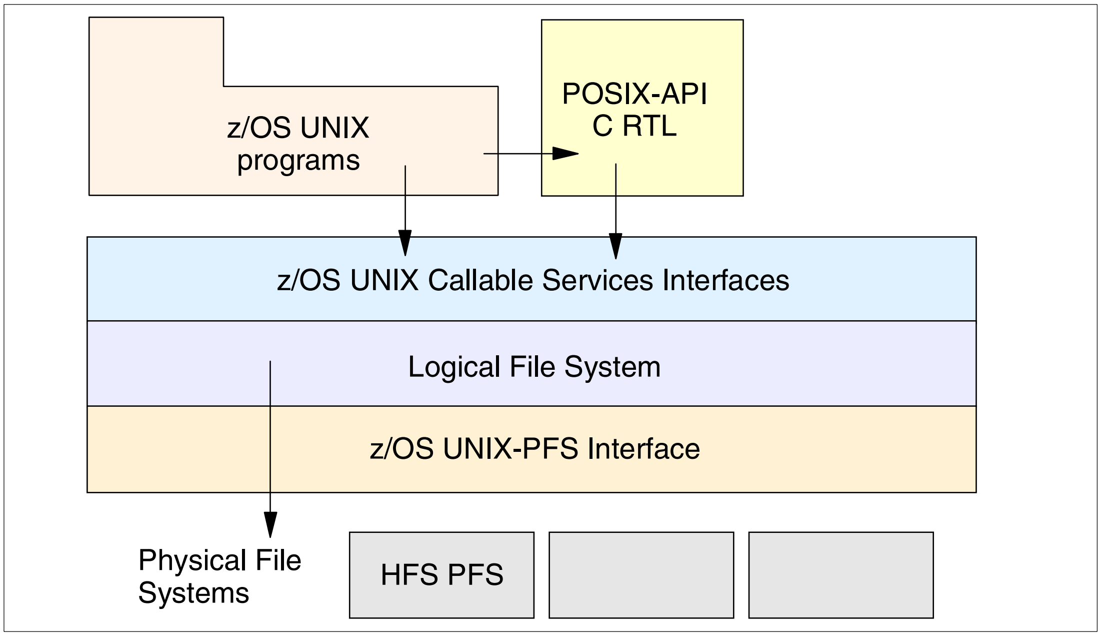


Figure 6-7 UNIX System Services

In a UNIX System Services environment, the physical file systems are defined in the BPXPRMxx parmlib member. The zFS physical file system is also to be defined in the parmlib member. Figure 6-8 on page 113 shows all the physical file systems that can be defined in a UNIX System Services environment.

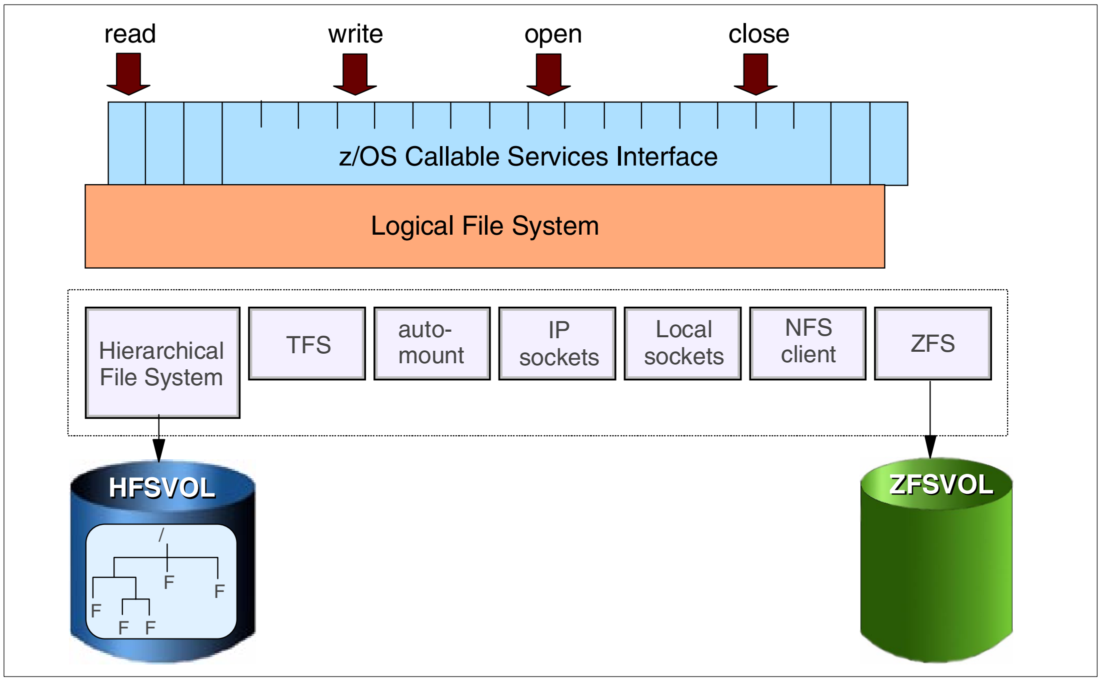


Figure 6-8 UNIX System Services physical file systems

6.5.5 zFS installation

With the Distributed File Service installed, if you only require zFS, the post-installation and configuration steps for the DFS client, DFS server, and the SMB server are not required to be performed. Also, the DFS client, DFS server, or SMB server do not need to be started to run zFS.

Note: The installation of DCE on the system is not required for zFS.

zFS executables

zFS executables, consisting of five PDS load modules, must reside in an APF-authorized library; also, there is one APF-authorized HFS executable.

The new zFS-specific executables in IOE.SIOELMOD are:

IOEZM001 Aggregate format utility program, alias IOEAGFMT.

IOEZM002 Aggregate salvage utility program, alias IOEAGSLV.

IOEZM003 Initialization and administration request handler. This program runs in the ZFS colony address space, alias IOEFSADM.

IOEZM004 Primary file request handler that runs in the ZFS colony address space, alias IOEFSCM.

IOEZM005 An administration command interface. This is an APF authorized HFS executable named zfsadm, alias IOEZADM.

IOEZM006 zFS trace formatter program for use under the direction of IBM in a customer installation. This is also an APF-authorized HFS executable named zfsadm.

6.5.6 Customization steps for zFS

To begin using the zFS file system, you must install the z/OS Distributed File Service base element since it is required for zFS. If the DFS support is already installed and configured on your system, then you only need to customize the new zFS file system. Perform the following steps to define and run zFS as a z/OS UNIX physical file system:

- zFS RACF definitions.
- Define zFS to UNIX System Services with the BPXPRMxx parmlib statement using the FILESYSTYPE entry.
  - Customize the ZFS procedure.
- Allocate and format a zFS aggregate: IOEAGFMT.
- Define the zFS parameter file for configuration options for aggregates: IOEFSPRM.
- Mount a zFS file system in the UNIX System Services directory hierarchy.
  - Mount compatibility mode aggregates.
    - MOUNT command - `AUTOMNT`.
- Attach a zFS multi-file system aggregate.
  - **`zfsadm`** command, or use IOEFSPRM member.
  - MOUNT command (TSO/E).
- Define a zFS file system in an aggregate.
- Mount a zFS file system.
  - Create a directory mount point - use `MOUNT` command.

- Access the zFS file system the same way you access an HFS file system.

6.5.7 zFS RACF definitions

In addition to the RACF commands documented for the DFS in the Program Directory for the release, the following RACF definitions are required for zFS:

```
RDEFINE STARTED ZFS.** STDATA(USER(DFS))
SETROPTS RACLIST(STARTED) REFRESH
```

A summary of all the RACF updates required to install the Distributed File Service and run zFS as a physical file system are shown in Figure 6-9.

```
ADDGROUP DFSGRP SUPGROUP(SYS1) OMVS(GID(2))
ADDUSER DFS OMVS(HOME(/opt/dfslocal/home/dfscntl) UID(0))
     DFLTGRP(DFSGRP) AUTHORITY(CREATE) UACC(NONE)
RDEFINE STARTED DFS.** STDATA(USER(DFS))
RDEFINE STARTED DFSCM.** STDATA(USER(DFS))
RDEFINE STARTED ZFS.** STDATA(USER(DFS))
   SETROPTS RACLIST(STARTED)
   SETROPTS RACLIST(STARTED) REFRESH
```

Figure 6-9 RACF definitions for Distributed File Service

> **Note:** A userid other than userid=DFS can be used to run the ZFS started task if it is defined with the same RACF characteristics as shown for the DFS user ID.

6.5.8 BPXPRMxx definitions

Here is an example FILESYSTYPE statement for the zFS physical file system. It includes an ASNAME parameter specifying the ZFS procedure to be started. The ASNAME is required for any PFS that runs in a colony address space.

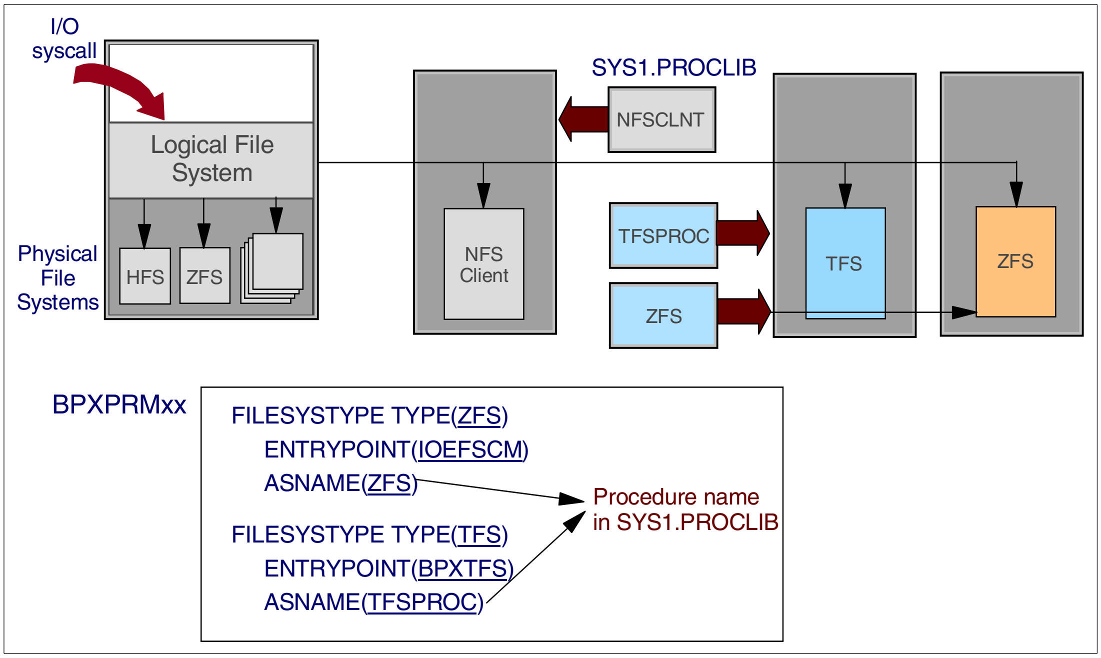
```
FILESYSTYPE TYPE(ZFS) ENTRYPOINT(IOEFSCM) ASNAME(ZFS)
```

As the aggregates are attached, you cannot have MOUNT statements for zFS file systems in the BPXPRMxx parmlib member. We recommend that you put mount commands in the /etc/rc file.

ZFS colony address space

zFS runs in a UNIX System Services colony address space. A colony address space is a separate address space, that is, separate from the UNIX System Services address space. HFS runs inside the UNIX System Services address space.

Whether or not a PFS runs in a colony address space is controlled by the ASNAME parameter in the FILESYSTYPE statement for the PFS in the BPXPRMxx member of the SYS1.PARMLIB concatenation.



```
FILESYSTYPE TYPE(ZFS)
   ENTRYPOINT(IOEFSCM)
   ASNAME(ZFS)

FILESYSTYPE TYPE(TFS)
   ENTRYPOINT(BPXTFS)
   ASNAME(TFSPROC)
```

Figure 6-10 zFS executes in a colony address space

JCL for zFS started task

The following JCL is required to start zFS. The file system runs in a colony address space. The JCL must be in the a proclib member, as shown in Figure 6-10, with the membername that is referenced by the ASNAME parameter the BPXPRMxx FILESYSTYPE for the zFS, as noted previously.

The IOEZPRM DD statement can be omitted from the JCL procedure, or the data set can exist with no parameters. It can be a sequential data set or a PDS member.

```
//ZFS      PROC REGSIZE=0M
//ZFSGO EXEC PGM=BPXVCLNY,REGION=&REGSIZE,TIME=1440
//*STEPLIB  DD DISP=SHR,DSN=IOE.SIOELMOD
//IOEZPRM DD DSN=hlq.PARMLIB(IOEFSPRM),DISP=SHR
// PEND
```

Figure 6-11 zFS proclib member to start zFS

Note: The DDNAME=IOEZPRM identifies the optional DD statement identifying the parameter file for zFS. If you accept all defaults, you do not need the parameter file.

Although the zFS parameter file defined in the JCL DD statement IOEZPRM is optional, we recommend that you use it during testing to identify the parameter file in use when testing zFS.

6.5.9 Create a zFS aggregate

A multi-file system or compatibility mode aggregate must have been defined before a zFS file system can be created in the aggregate.

A zFS aggregate is created by defining a VSAM linear data set and then formatting that VSAM LDS as an aggregate. This is done once for each zFS aggregate. You cannot assign more than one VSAM LDS per aggregate. An aggregate can contain one or more zFS file systems. A zFS file system is equivalent to an HFS file system.

The VSAM LDS is allocated with the VSAM utility program IDCAMS, as shown in Figure 6-12. The JCL shows the allocation of both types of aggregates.

Note: Multiple volumes may be specified when you define the aggregate.

```
//RFRZAL  JOB (999,POK),'R F',CLASS=A,MSGCLASS=U,NOTIFY=&SYSUID,
//        REGION=0M
//STEP1   EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=*
//DISK     DD  DISP=OLD,UNIT=3390,VOL=SER=ZFSVOL
//SYSIN    DD  *
  DEFINE CLUSTER  -
         (NAME (OMVS.PAY.ZFS) VOL(ZFSVOL) -
         LINEAR CYL(200 0) SHAREOPTIONS(2))
  DEFINE CLUSTER  -
         (NAME (OMVS.WEB.ZFS) VOL(ZFSVOL) -
         LINEAR CYL(200 0) SHAREOPTIONS(2))
```

Figure 6-12   Defining compatibility mode and multi-file system aggregates

Note: Allocation of a zFS aggregate or file system cannot be done from the ISHELL in this release.

Managing space on volumes

The VSAM LDS used for a zFS aggregate can be defined to multiple volumes. If the VSAM LDS is extended to additional volumes after the aggregate is initially formatted, the aggregate must be formatted to extend to the additional volume using the **`zfsadm grow`** command, which uses the IOEZADM utility.

Note: zFS aggregates can be greater than 4 GB. You can create an aggregate greater than 4 GB only by using an extended format and addressability in the data class of the data set.

6.5.10 IOEAGFMT format utility

The IOEAGFMT utility formats multi-file system aggregates and compatibility mode aggregates. It is a stand-alone utility that does not require zFS to be active. No entries are required in the IOEFSPRM configuration file to run IOEAGFMT.

These are the parameters, shown in Figure 6-1 on page 117, that can be specified in the IOEAGFMT utility. Most of these parameters are self explanatory.

Table 6-1   IOEAGFMT format utility parameters

Parameter	Values	Description
-aggregate	VSAM LDS cluster name	The name of the data set to format.
-blocksize	4K, 8K, 16K, 32K, 64K	The size in bytes of the logical block size.

Parameter	Values	Description
-fragsize	1K, 2K, ..., up to blocksize	The size in bytes of a fragment. Multiple files can share a logical block if they are less than (logical block size - fragment size).
-aggrsize	A number	The size in blocks of the aggregate. The default is the number of blocks that will fit in the primary allocation.
-logsize	A number	The size in blocks of the log. The default is 1% of the aggrsize.
-overwrite	N/A	Reformat an existing aggregate.
-compat	N/A	Create a compatibility mode aggregate.

Note: A fragment is part of a block that can hold a file that is smaller than a block. This allows zFS to store multiple (small) files in a block so that it doesn't waste too much DASD space for small files.

All zFS aggregates must be formatted before use, including HFS compatibility zFS aggregates. The HFS compatibility zFS aggregates do not need to be attached.

A compatibility mode aggregate is formatted with this JCL:

```
//RFRAGF  JOB (999,POK),'R F',CLASS=A,MSGCLASS=U,NOTIFY=&SYSUID,
// REGION=0M
//STEP1   EXEC PGM=IOEAGFMT,
//          PARM=(' -aggregate omvs.pay.zfs -compat ')
//SYSPRINT DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//STDERR   DD  SYSOUT=*
//CEEDUMP  DD  SYSOUT=*
```

A multi-file system aggregate is formatted with this JCL:

```
//RFRAGF  JOB (999,POK),'R F',CLASS=A,MSGCLASS=U,NOTIFY=&SYSUID,
// REGION=0M
//STEP1   EXEC PGM=IOEAGFMT,
//          PARM=(' -aggregate omvs.web.zfs ')
//SYSPRINT DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//STDERR   DD  SYSOUT=*
//CEEDUMP  DD  SYSOUT=*
```

Figure 6-13 Using IOEAGFMT utility to format the aggregates

Attention: The PARM= parameters must be in lower case in the JCL.

IOEAGFMT job completion messages

To determine if the aggregates were successfully defined, look for the messages shown in Figure 6-14 on page 119.

```
IOEZ00004I Loading dataset 'omvs.pay.zfs'.
IOEZ00005I Dataset 'omvs.pay.zfs' loaded successfully.
*** Using default initialempty value of 1.
*** Using default number of (8192-byte) blocks: 17999
*** Defaulting to 179 log blocks(maximum of 19 concurrent transactions).
Done.  /dev/lfs1/omvs.pay.zfs is now an zFS aggregate.
IOEZ00071I Attaching aggregate OMVS.PAY.ZFS to create hfs-compatible file system
IOEZ00074I Creating file system of size 140487K, owner id 0, group id 2, permissions x1ED
IOEZ00048I Detaching aggregate OMVS.PAY.ZFS
IOEZ00077I HFS-compatibility aggregate OMVS.PAY.ZFS has been successfully created
IOEZ00004I Loading dataset 'omvs.web.zfs'.
IOEZ00005I Dataset 'omvs.web.zfs' loaded successfully.
*** Using default initialempty value of 1.
*** Using default number of (8192-byte) blocks: 17999
*** Defaulting to 179 log blocks(maximum of 19 concurrent transactions).
Done.  /dev/lfs1/omvs.web.zfs is now an zFS aggregate.
```

Figure 6-14 Job completion messages from the format job

6.5.11 The zFS parameter file

The zFS parameter file, IOEFSPRM, is an optional file that can contain parameters for controlling the following:

- zFS operation
- Directing message output from zFS
- Entries for multiple file system aggregates

The file is defined with a DD statement in the ZFS procedure. The file can exist, with no parameters. Parameters are only required if you want to override the defaults for the zFS parameters. The file can be a PDS member or a sequential file. A sample IOEFSPRM file will be supplied in IOE.SIOESAMP(IOEFSPRM). Table 6-2 describes the options you can specify in the IOEFSPRM file.

Table 6-2 IOEFSPRM file options

Option	Values	Description
adm_threads	n, 10	The number of threads to handle pfsctl or mount requests.
auto_attach	ON, OFF	Whether aggregates in IOEFSPRM are automatically attached at start-up.
user_cache_size	n, 256M	The size of the cache used to contain file data.
meta_cache_size	n, 32M	The size of the cache used to contain metadata.
log_cache_size	n, 64M	The size of the cache used to contain log buffers.
sync_interval	n, 30	The number of seconds between syncs.
vnode_cache_size	n, 8192	The initial size of the internal zFS vnode cache.
nbs	ON, OFF	New Block Security. A global specification of whether we guarantee that new block data that has not made it to the disk before a crash will be shown as binary zeros after the crash.
aggrfull	(n,m), OFF	The threshold and increment for reporting aggregate full msgs to the op.

Option	Values	Description
fsfull	(n,m), OFF	The threshold and increment for reporting file system full msgs to the op.

IOEFSPRM parameters for attach

The zFS parameter file contains information needed during zFS aggregate attach.

```
define_aggr [R/O | R/W] [[no]attach] [[no]nbs] [aggrfull(x,y)]
            cluster(VSAM_LDS_cluster_name)
```

where:

R/O The entire aggregate (all file systems) are R/O.

attach The aggregate is attached at PFS start up.

nbs Using new block security means an allocated block that has no data written into it, is shown as binary zeros (after a crash).

aggrfull The threshold and increment percentages for writing aggregate full error messages to the operator.

msg_output The zFS messages are directed to the system console and/or system console log if this parameter is not specified.

An asterisk (*) in column 1 indicates a comment line in the file. Also, blanks lines are allowed in the file.

6.5.12 zfsadm command

This command can be used to manage file systems and aggregates. The **zfsadm** command can be run as a shell command from the z/OS UNIX System Services shell (OMVS). Both the command and the program require the zFS colony address space to be up and running.

zFS file systems can be created using JCL, shown in Figure 6-16 on page 122, or by using the **zfsadm** command.

zfsadm subcommands

The **zfsadm** command or the IOEZADM program has the following subcommands:

zfsadm attach	Attach an aggregate
zfsadm apropos	Display first line of help entry
zfsadm detach	Detach an aggregate
zfsadm grow	Grow an aggregate
zfsadm aggrinfo	Obtain information on an attached aggregate
zfsadm clone	Clone a filesystem
zfsadm clonesys	Clone multiple filesystems
zfsadm create	Create a filesystem
zfsadm delete	Delete a filesystem
zfsadm help	Get help on commands
zfsadm lsaggr	List aggregates
zfsadm lsfs	List filesystem information
zfsadm lsquota	List filesystem information
zfsadm quiesce	Quiesce an aggregate
zfsadm rename	Rename a filesystem
zfsadm setquota	Set filesystem quota
zfsadm unquiesce	Unquiesce an aggregate

6.5.13 Attaching an aggregate

Compatibility mode aggregates do not need to be attached.

A multi-file system aggregate must be attached before issuing a mount command for it and before a zFS file system can be created in the aggregate.There are three ways to attach a multi-file system aggregate. The attach can be done:

- On the zFS colony address space startup by having an entry in the IOEFSPRM file
- Using the IOEZADM program in a submitted job
- Using the **`zfsadm attach`** command

> **Note:** You do not have to attach an HFS compatibility mode zFS aggregate. You need only to mount or automount the file system.

IOEFSPRM file

If you have created and formatted a zFS multi-file system aggregate, you may add an entry in the IOEFSPRM file for the aggregate. This causes the multi-file system aggregate to be attached when zFS is started. Add the following line to the IOEFSPRM file:

```
define_aggr R/W attach cluster(OMVS.WEB.ZFS)
```

IOEZADM program

If a zFS multiple file system aggregate is created after ZFS is started, you must attach the zFS aggregate to tell the zFS PFS about it. Use the **`zfsadm`** command, discussed in the next section, or use the JCL shown in Figure 6-15, to attach an aggregate. A sample of the JCL to run PGM=IOEZADM is supplied in IOE.SIOESAMP(IOEZADM).

```
//ZFSATTA JOB ,'ZFS Attach Aggregate',
//         CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//ZFSADMA  EXEC   PGM=IOEZADM,REGION=0M,
// PARM=('attach -aggregate OMVS.WEB.ZFS')
//STEPLIB  DD DISP=SHR,DSN=IOE.SIOELMOD
//SYSPRINT DD   SYSOUT=T
//STDOUT   DD   SYSOUT=T
//STDERR   DD   SYSOUT=T
//SYSUDUMP DD   SYSOUT=T
//CEEDUMP  DD   SYSOUT=T
//*
```

Figure 6-15 JCL to attach a multi-file system aggregate

zfsadm command for attach

Verify that during your z/OS Distributed File Service installation a symbolic link has been created for access to the **`zfsadm`** command. If for some reason this has not been done, issue the following command to create a symbolic link in the /bin directory:

```
ln -s /usr/lpp/dfs/global/bin/zfsadm /bin/zfsadm
```

The following command can be used from the OMVS shell to attach a multi-file aggregate:

```
zfsadm attach -aggregate omvs.web.zfs -aggrfull 90,5 -nbs
```

You should receive the following message for the attach:

```
IOEZ00117I Aggregate OMVS.WEB.ZFS attached successfully
```

Note: The aggregate name specified in the command is not case sensitive; it will be translated to upper case if entered in lower case.

6.5.14 Defining a zFS file system

There are two types of aggregates that zFS file systems can be defined in, as follows:

Compatibility mode There can be only one zFS file system in a compatibility mode aggregate. The name of the file system is the same as the aggregate name. The file system size is the size of the VSAM LDS aggregate. Use this file system name to mount the file system.

Multi-file mode A multi-file system aggregate can contain multiple zFS file systems. The aggregate has a name separate from the file system names. Each file system has a predefined size. A multi-file aggregate must be attached before a zFS file system can be defined in it.

Multi-file system aggregates

Once an aggregate has been attached, a zFS file system can be created in the aggregate. The zFS colony address space must be running. There are two ways to define a zFS file system, as follows:

- Use the IOEZADM program using JCL.
- Use the `zfsadm` command from the OMVS shell.

IOEZADM program

To define a zFS file system in an aggregate, use the IOEZADM program with the subcommand ‘create’. When defining the parm= keyword, as shown in Figure 6-16, specify the following:

- The file system name, which is case sensitive
- The aggregate name where the file system is to be defined
- The size of the file system in 1K blocks, which is a required parameter

Note: There is no default logical size for a zFS file system. The zFS file system name is case sensitive, and must be upper case.

```
//ZFZADM   JOB ,'ZFS Create Filesys',NOTIFY=ROGERS,
//         CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1),TIME=1440
//*
//ZFSADM   EXEC   PGM=IOEZADM,REGION=0M,
//         PARM=('create -filesystem ZFSFS01
//             -aggregate omvs.web.zfs -size 5000')
//STEPLIB DD DISP=SHR,DSN=IOE.SIOELMOD
//SYSPRINT DD   SYSOUT=T
//STDOUT   DD   SYSOUT=T
//STDERR   DD   SYSOUT=T
//SYSUDUMP DD   SYSOUT=T
//CEEDUMP  DD   SYSOUT=T
//*
```

Figure 6-16 Defining a zFS file system in a specific aggregate

zfsadm command

If you prefer to use the OMVS shell to define the zFS file system using the **zfsadm** command, the following command is equivalent to the JCL example above.

```
zfsadm create -filesystem ZFSFS01 -size 5000 -aggregate OMVS.WEB.ZFS
```

Look for the following message for the zfsadm command:

```
IOEZ00099I File system ZFS001 created successfully
```

6.5.15 Mounting zFS file systems

Once the file systems have been defined, they need to be mounted at a directory mount point for users to access the file system. zFS file systems can be automatically mounted at IPL by specifying them in the /etc/rc file using an OMVS shell mount command, **/usr/sbin/mount**.

There are two types of aggregates that contain file systems to be mounted, as follows:

Compatibility mode A zFS file system in a compatibility mode aggregate can be mounted, using the **mount** command, at an installation-created mount point. A zFS file system in a compatibility mode aggregate also can be AUTOMOVEed or automounted using the automount facility.

Multi-file mode A multi-file system aggregate must be attached before a zFS file system can be created. See “Attaching an aggregate” on page 121. After creating a directory for the mount point, you may use the OMVS **mount** command for the mount.

Restriction: A BPXPRMxx MOUNT statement cannot be used to mount a zFS file system in this release.

Creating a mount point

The OMVS shell or TSO/E may be used to issue the **MOUNT** command. The compatibility mode zFS file system can also be mounted using the following options:

- AUTOMOVE
- Automount facility

Compatibility mode mount

Choose a directory mount point in your UNIX System Services root directory structure to mount the zFS file system. Issue the following commands to create the mount point /etc/zfs:

```
cd /etc/
mkdir zfs
```

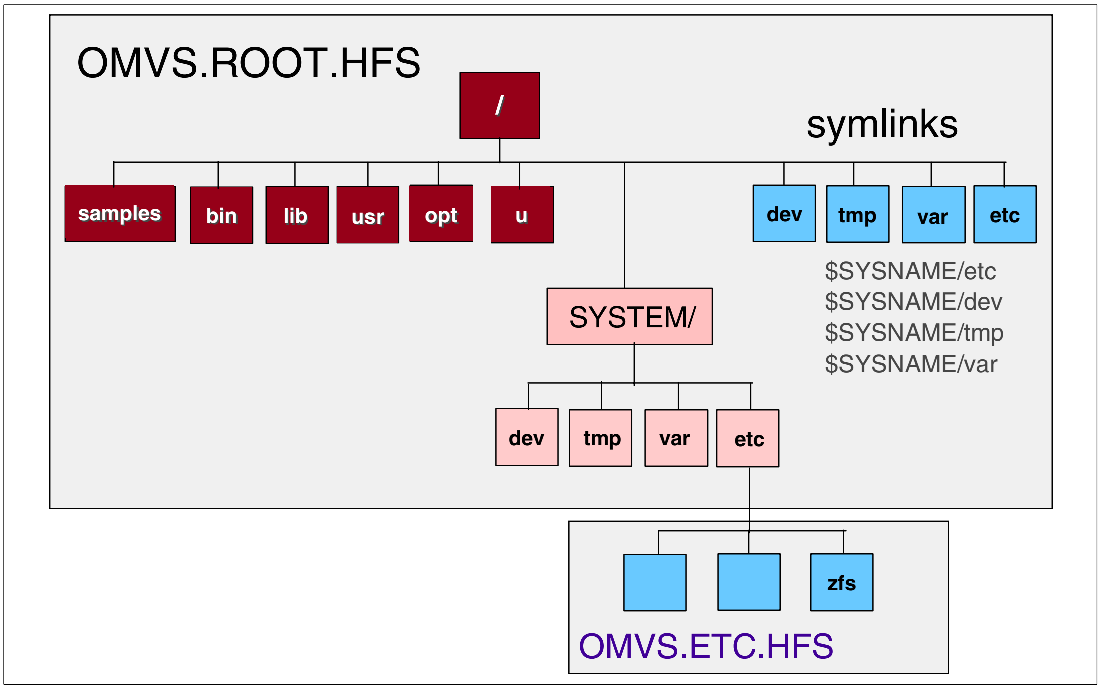


Figure 6-17 Creating the ZFS directory in the root file system

In our example in Figure 6-18 on page 125, we are showing multiple file systems mounted at specified mount points. Continue with the example as follows:

```
cd zfs
mkdir zfsc
```

Issue the **MOUNT** command from TSO/E for the compatibility mode file system as follows:

```
MOUNT FILESYSTEM(‘OMVS.PAY.ZFS’) TYPE(ZFS) MODE(RDWR) MOUNTPOINT('/etc/zfs/zfsc')
```

Multiple file mode mounts

From /etc/zfs, issue the following:

```
mkdir zfs1
mkdir zfs2
mkdir zfs3
```

Issue the **MOUNT** command from TSO/E for the multi-file file system as follows:

```
MOUNT  FILESYSTEM('ZFSFS01') TYPE(ZFS) MODE(RDWR) MOUNTPOINT('/etc/zfs/zfs1')
MOUNT  FILESYSTEM('ZFSFS02') TYPE(ZFS) MODE(RDWR) MOUNTPOINT('/etc/zfs/zfs2')
MOUNT  FILESYSTEM('ZFSFS03') TYPE(ZFS) MODE(RDWR) MOUNTPOINT('/etc/zfs/zfs3')
```

You can place an OMVS shell mount command in the /etc/rc file to mount a zFS file system when UNIX System Services is started, as follows:

```
/usr/sbin/mount -t ZFS -f ZFSFS01 /etc/zfs/zfs1
```

As shown in Figure 6-18, there are two multiple file aggregates, one with one file system in it and the other with two file systems. The compat mode aggregate, of course, has only one file system.

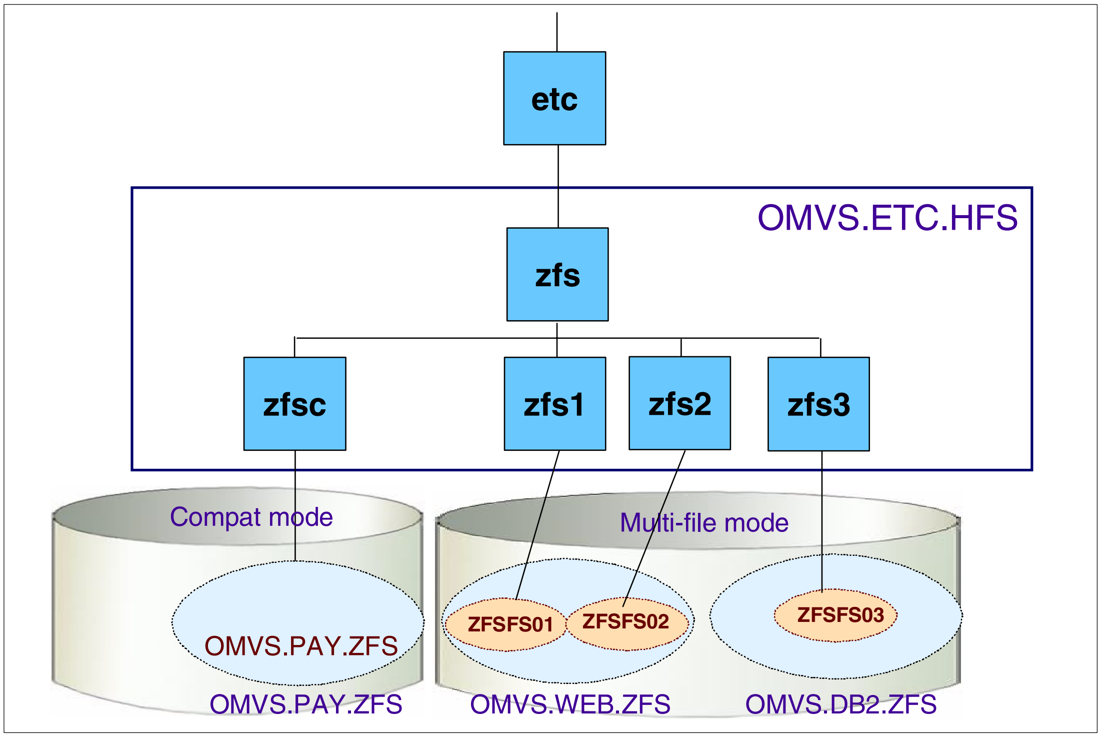


Figure 6-18 Mounted zFS file systems

6.5.16 zFS and sysplex considerations

zFS file systems can be shared in a sysplex, but this is restricted to file systems that are compatibility mode aggregates only. zFS file systems in multi-file aggregates cannot be shared.

Note: Multiple file mode aggregates file system should be mounted NOAUTOMOVE.

There is also an implication for automounted file systems. zFS file systems in compatibility mode aggregates can be automounted in a sysplex. zFS file systems in a multi-file system aggregate cannot be automounted in a sysplex.

6.5.17 zFS recovery

zFS is a logging file system. It logs metadata updates. On a system crash, the log is replayed to bring the file system to a consistent state. zFS only needs to examine and replay its log (much smaller than the entire file system) to ensure consistency.

Metadata updates

zFS uses its log of metadata updates to reconstruct the file system to consistency if the system fails and the file system is restarted.

I/O requests

I/O requests are started immediately so that if a system crash occurs, most data is already on disk. When an application writes a block of data to zFS, an I/O request to the DASD is started immediately, but it is also asynchronous and the requestor doesn't wait for the I/O to complete. This means that more data will be on the disk in the event of a crash as compared to HFS support, which normally only writes data to DASD at synchronous intervals.

zFS backup and restore

zFS aggregates must be quiesced before being backed up. The quiesce of the aggregate is accomplished using the **`zfsadm quiesce`** command.

6.5.18 Restrictions

The following restrictions are known at the time of writing:

- A zFS file system cannot be mounted as the root file system.
- You cannot mount a zFS from MOUNT statements in BPXPRMxx.
- Allocation of a zFS aggregate cannot be done from the ISHELL.
- A zFS file system should not be used in place of any HFS file system used as the z/OS installation file system, such as /etc/HFS.

Multiple file system aggregates specified in the zFS parameter file will not be mounted when the zFS address space initializes. Multiple file system aggregates must be specified in the /etc/rc file using a **`mount`** command.

For additional information on zFS, see *z/OS Distributed File Service DFS Administration*, SC24-5915 and *z/OS Distributed File Service Message and Codes*, SC24-5917.

6.6 zFS availability

zFS code is not available on the initial order of z/OS Version 1 Release 2. For the enablement of the zFS file system for General Availability, install the following APARs/PTFs:

- OW50850 / UW82925
- OW51563 / UW83377

The support for zFS on Releases OS/390 2.10 and z/OS 1.1 is provided by APAR OW51780.

7

UNIX System Services

z/OS UNIX System Services in z/OS V1R2 includes new functions and enhancements in the following areas:

- Enhanced ASCII functionality
- Shell and utility enhancements
- System management commands
- Logical file system
- HFS program control

In this chapter we describe each of these features in detail.

7.1 Enhanced ASCII functionality

The enhanced ASCII functionality allows z/OS UNIX System Services to deal with files that are in both ASCII and EBCDIC format. The enhanced ASCII support makes it easier to port applications developed on ASCII platforms to z/OS platforms by providing a limited ASCII-to-EBCDIC translation and vice versa. ASCII uses codepage ISO8859-1, while z/OS UNIX System Services by default uses the EBCDIC codepage IBM-1047. The enhanced ASCII functionality is limited to these two codepages.

The solution includes:

- Automatic codeset conversion (Autoconversion)
  - File tagging
- C/C++ Compiler options
- Language environment runtime library (RTL) enhancements

We recommend that you limit the enabling of autoconversion to the smallest environment possible. It is important to understand that file tagging and autoconversion are independent operations. You can tag files without enabling autoconversion and vice versa.

7.1.1 ASCII support overview

z/OS and the zSeries processor are an EBCDIC platform. Extended Binary-Coded-Decimal Interchange Code (EBCDIC) is an eight-bit code that gives 256 possible combinations and was originally developed for the IBM S/360. Today EBCDIC is used by the zSeries and iSeries (formerly AS/400) of IBM servers. This means that the zSeries processors have programs that are compiled to handle EBCDIC data, EBCDIC encoding of characters, and devices that are configured for EBCDIC. UNIX applications on other platforms are encoded in the ASCII code set. This includes programs, literal strings within programs, and data.

Before this support, application programs that ran under z/OS UNIX System Services had to be compiled in EBCDIC format, and they expected data encoded in EBCDIC. The **iconv** command was available to convert files from ASCII to EBCDIC. Enhanced ASCII introduces automatic conversion of data between ASCII and EBCDIC.

If you implement the new enhanced ASCII support, the EBCDIC nature of the z/OS platform remains. However, if you compile your C program as ASCII, the EBCDIC nature of the z/OS platform can be partially hidden.

This support is limited to z/OS UNIX files. It does not apply to MVS data sets, even though they can be accessed by z/OS UNIX System Services.

New ASCII support

This support is introduced to help the porting of ASCII applications to the z/OS platform. Specifically, the support provides the ability to:

- Build ASCII-based applications by producing object code with ASCII string literals and character constants and a flag that identifies applications as ASCII or EBCDIC.
- Use Unicode-based wide characters (wchar_t) in ASCII-based applications.
- Transparently call native ASCII run time library functions from ASCII-based applications.
- Process user-defined ASCII multi-byte code pages with user-supplied code set-related methods.

- Create ASCII-based local objects which allow processing of ASCII data natively at run time.

There are several ways to read or write to a file. The callable services BPX1RED and BPX1WRT, which are the ones used when **read()**, **write()**, **pread()**, and **pwrite()** are issued, are the only services supported for automatic conversion. Only regular files, and named and unnamed pipes are supported. Socket and directory files are not supported.

7.2 Automatic conversion

Figure 7-1 shows a program that is reading or writing from and to an ASCII file. The program does not need to know that the file is in ASCII; the Logical File System (LFS) checks and finds that it is okay to perform the automatic conversion. Nothing indicates that an automatic conversion has occurred. You can use **Ctrace** to find it out.

Automatic conversion of files from ASCII to EBCDIC and vise versa is controlled globally by the AUTOCVT(ON) or (OFF) statement in the BPXPRMxx parmlib member. AUTOCVT can be overridden locally by the individual programs at thread level.

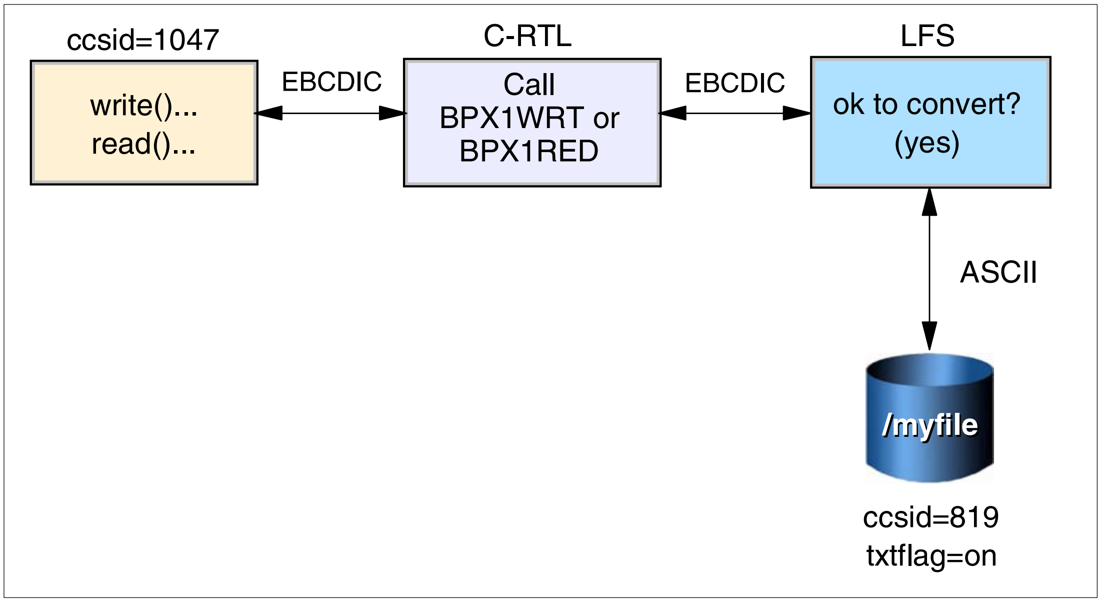


Figure 7-1 Automatic conversion of an ASCII file

The coded character set identifier (CCSID) for enhanced ASCII functionality is a 16-bit value, a number that represents a character set used by file tagging. It identifies the current character set of text strings within a program. This is stored in the file tag of new files or used for the automatic conversion of old files when autoconversion is in effect.

Note: If AUTOCVT is set to enable enhanced ASCII, the performance of your z/OS UNIX environment is affected because every read and write operation for a file must be checked. That means all HFS file systems and every file in the file system is checked to see if conversion is necessary. It is recommended that you use AUTOCVT(OFF). If you want to enable enhanced ASCII using another method, see “File tagging overview” on page 132.

7.2.1 Autoconversion

The enhanced ASCII functionality provides a limited automatic conversion (autoconversion) from ASCII to EBCDIC and vice versa. There may be many reasons why you do not want to use autoconversion. We recommend that you limit the enabling of autoconversion to the smallest possible environment available.

There are several conditions that must be met before autoconversion is done:

- The autoconversion must be activated by any of the following:
  - A parameter in parmlib member BPXPRMxx
  - An environment variable
  - A runtime option
- The file must be tagged with a txtflag on and contain a valid codeset
- The program codeset must be different from the file tag codeset

Programs and files

Automatic conversion is accomplished between programs and files that are tagged with different CCSIDs when a conversion table exists for that CCSID pair in the system.

The autoconversion is only supported between the codesets ISO8859-1 (ASCII) and IBM-1047 (z/OS UNIX System Services EBCDIC). These are the only codesets supported, no other converters exist.

Commands

Most commands that perform file I/O expect text data, so they allow autoconversion.

The autoconversion is controlled globally by the BPXPRMxx parmlib statement:

```
AUTOCVT(ON|OFF)
```

This statement activates or deactivates automatic conversion of text data files using CCSIDs for the program and its associated files. The CCSIDs are specified by the program or by setting the appropriate environment variables at run time. The system AUTOCVT indicator set by this statement can be overridden by individual programs at the thread level. You can think of AUTOCVT as a controlling switch only for existing programs that do not explicitly establish their own conversion environment.

AUTOCVT(OFF) This option deactivates autoconversion, and is the default value.

AUTOCVT(ON) When this is set, every read and write operation for a file must be checked to see if conversion is necessary. Thus, there is a performance penalty involved, even if no conversion occurs. Therefore, we recommend that you keep AUTOCVT(OFF) and have each program enabled, if possible, for conversion. To override the AUTOCVT setting, use the compile or C run-time environment variables that control conversion or issue **vacant()** in your program. The two new **vacant()** subcommands, F_CONTROL_CVT and F_SETTAG, are supported as both C run-time library variables and as callable services.

You can use the **SETOMVS** or **SET OMVS** commands to change the value of AUTOCVT between ON and OFF. Changing this conversion mode does not affect conversion of opened files for which I/O has already started.

7.2.2 Scope of autoconversion

Automatic conversion can be controlled at different environmental levels.

At the highest level, AUTOCVT(ON) can be specified in the BPXPRMxx parmlib member to enable automatic conversion for the entire z/OS UNIX System Services environment.

At the lowest level, a program can use the new subcommand of **`fcntl()`**, F_CONTROL_CVT to enable autoconversion for a single open file or enable automatic conversion using an environment variable _ BPXK_AUTOCVT, or the FILETAG run-time option, or both. When either of these options is used, it overrides the AUTOCVT system setting.

Autoconversion can also be controlled individually by a single program by using one of the following flags in the thread Thli control block:

- ThliCvtOn - Activates autoconversion for this thread.
- ThliCvtOff - Deactivates autoconversion for this thread.

Checks for autoconversion

The following checks are made to determine if autoconversion should be done:

- Is the environment enabled for conversion?
- Is there a file tag that indicates that the file is a candidate for autoconversion?
- Is the program CCSID different from the CCSID in the file tag?

All three checks must be true for the conversion to be done.

Figure 7-2 shows the different options you can use, and that must be fulfilled in order for autoconversion to take place.

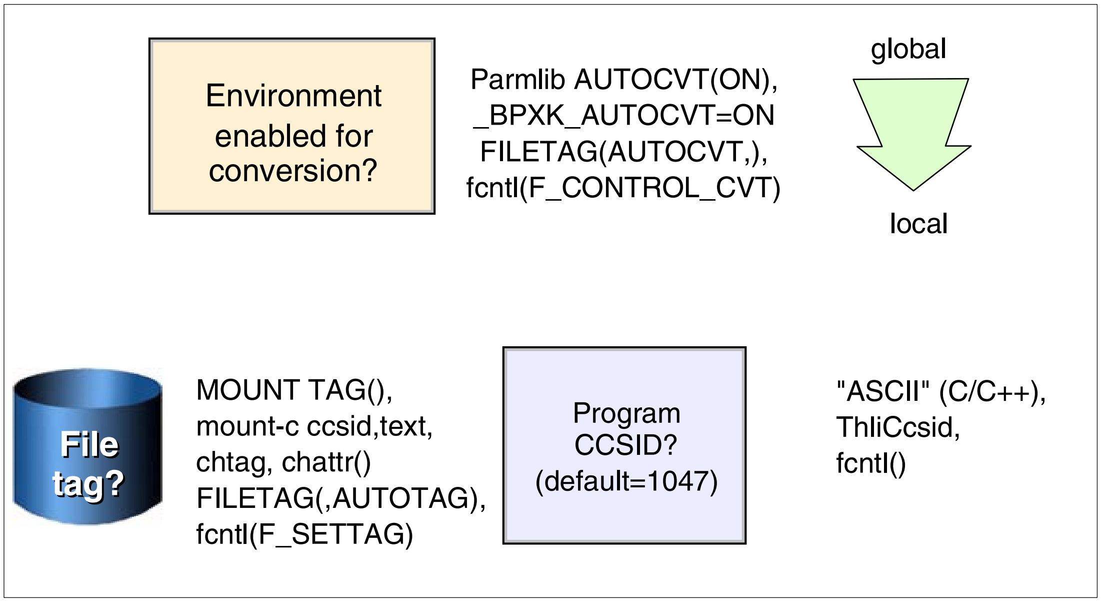


Figure 7-2 Scope of autoconversion

7.3 File tagging overview

To complement enhanced ASCII, support for file tagging is also provided. File tags are a way to identify the code set of the text data within files. A file tag is metadata associated with a file; you can think of the file tag as a yellow sticky note on the file containing information about the encoding character set used to write the data in the file.

File tagging and enabling automatic conversion are independent operations. You can tag files without enabling automatic conversion, and vice versa. Remember that enabling automatic conversion for the entire system by using the AUTOCVT statement in the BPXPRMxx member, means that every tagged file becomes subject to conversion by any program that reads from or writes to those tagged files. Because of this, it is possible to have checking done on many tagged files without any conversion occurring.

You should consider enabling automatic conversion in the smallest environment possible by using one of the following methods:

- _BPXK_AUTOCVT environment variable in a .profile
- The FILETAG run-time option

7.3.1 File tag metadata

File tags are used during automatic codeset conversion, but they are independent from autoconversion.

The file tag metadata contains two fields:

txtflag The txtflag indicates whether a file contains uniformly encoded text data or not. The txtflag can be either on or off.

When the txtflag is on, it indicates that the file is a textfile, and uniformly encoded in one specific character codeset, either ASCII or EBCDIC. This means that this file is a candidate for autoconversion. Only files with the txtflag on and a valid codeset are candidates for automatic conversion.

When the txtflag is off, it means the file content is non-uniformly encoded, and that the file is not a candidate for autoconversion.

You can tag your files without using autoconversion, use the filetag for your own information, and perform codeset conversion yourself.

codeset The 16 bit codeset value indicates which code page the data in the file is encoded in. When you tag a file, you can use a character code set name known to the system, or you can use the numeric value called coded character set ID (CCSID). This is a value between 0 and 65536. However, when TEXT is specified, the values of 0 and 65536 are illegal because those values imply no conversion. Other than this, the value is not checked as being valid and the corresponding code page is not checked as being installed.

If a numeric codeset name exists, the CCSID associated with that name will be used. The CCSID values that are associated with names are: 819, which means code page ISO8859-1; and 1047, which means code page IBM-1047. The CCSID can be used both for uniformly encoded text files and for files that contain mixed text and binary data; however, the latter files are not candidates for autoconversion.

The Program CCSID indicates the codeset expected by a C program. Only the values 819 and 1047 are supported. All processes and threads have a default program CCSID of 1047 (EBCDIC). When compiled with the ASCII option, the program CCSID defaults to 819.

7.3.2 How to tag files

There are several ways to tag single files or all files in a filesystem:

- The TAG parameter in BPXPRMxx
- The **chtag** shell command
- Enhancements to existing shell commands

BPXPRMxx TAG parameter

The ROOT and MOUNT statements in BPXPRMxx parmlib member have a new parameter named TAG. This parameter specifies whether implicit file tags are assigned to untagged files in the mounted file system. File tagging controls whether a file's data can be converted during file reading and writing.

The format of the TAG parameter is:

```
TAG (NOTEXT | TEXT, CCSID)
```

NOTEXT Specifies that none of the files in the file system will be automatically converted during file reading and writing. This is the default value.

TEXT Specifies that each untagged file is implicitly marked as containing pure text data. These files in the file system can be converted by autoconversion.

CCSID Specifies the option names for the coded character set identifier to be implicitly set for the untagged file.

> **Note:** Either TEXT or NOTEXT, and CCSID, must be specified when TAG is specified.

The tag itself becomes part of the metadata associated with the file, which means that the tag is not permanently stored with the file. The tag is only associated with the file during reading and writing, or when stat()-type functions are issued. The TAG parameter applies to all files in the file system which do not have a file tag. Any file without a tag is implicitly assigned the file tag from the **mount** command. The file tag is no longer there when the file system is unmounted.

Files created in the file system after the mount occurs are also implicitly tagged. For example, if you have a file system mount as ASCII text, you can use OEDIT or vi to create an ASCII text file. Even though OEDIT and vi are EBCDIC programs, autoconversion translates the file to ASCII.

Tag parameter examples

Here are some examples of using the TAG parameter:

TAG(TEXT,819)	Identifies text file containing ASCII (ISO-8859–1) data.
TAG(TEXT,1047)	Identifies text file containing EBCDIC ((IBM-1047) data.
TAG(NOTEXT,65536)	Tags file as containing binary or unknown data.
TAG(NOTEXT,0)	Is the equivalent of not specifying the TAG parameter.
TAG(NOTEXT,273)	Tags file with the German code set (IBM-273), but the file is ineligible for automatic conversion.

7.3.3 Shell commands for tags

There is a new shell command, **chtag**, and enhancements to other existing shell commands to support file tagging as follows:

- **chtag**
- **ls**
- **cp**
- **iconv**

chtag command

With the new shell command **chtag,** you can tag files, or change or display information in the file tag. You must have write permission to the file or be a superuser to use the **chtag** command. The format of the command is:

```
chtag -tc codeset file
```

This command tags the specified file as a textfile uniformly encoded in the specified codeset. The file becomes a candidate for autoconversion.

If you want to tag all files in a directory and its subdirectories, you can use the **-R** parameter on the command. The pathname must be a directory. The format of this command is:

```
chtag -tc codeset -R dir
```

-t Indicates that the specified file contains pure text data and, if used alone, sets txtflag=ON.

-c Sets or changes the codeset. Files that are tagged with this option and contain a valid codeset are candidates for automatic conversion.

-R Changes the file tag information on all of the files and subdirectories under that directory.

You can tag the file as a file that contains only binary data. The **-b** option sets the txtflag off, and the file is not a candidate for autoconversion. The format of this command is:

```
chtag -b mypgm
```

You can remove the tag from a file by using the **-r** option. The format of this command is:

```
chtag -r file
```

You can display the file tags by using the **-p** option. The format of this command is:

```
chtag -p file
```

ls command

The **ls** command has a new parameter, **-T**, that can be used to display file tags. This parameter can be combined with other **ls** parameters. The tag information output is the same format as the **chtag -p** command. You can see output from the list command in Figure 7-3 on page 135.

cp command

The **cp** command has been enhanced to support file tagging. The format of the command is:

```
cp -O c=codeset source target
```

You can use the **-O** parameter to tag the target file in the copy operation as a text file encoded in the specified code set, as specified with the c= option. Figure 7-3 shows how to copy the file data to the file ebcd and tag the new ebcd file as a text file encoded in EBCDIC codepage 1047. The second command lists the directory with the **-T** parameter on the command to see the file tags.

```
FRAMHUS@SC59:/u/framhus> cp -O c=IBM-1047 data ebcd
FRAMHUS@SC59:/u/framhus> ls -lT
total 16
- untagged    T=off -rw-r--r--   1 STC      SYS1         115 Jun 13 13:25 data
t IBM-1047    T=on  -rw-r--r--   1 STC      SYS1         115 Jun 13 13:30 ebcd
FRAMHUS@SC59:/u/framhus>
```

Figure 7-3 Copy and tag a file

iconv command

The **iconv** command converts characters in a file from one codepage to another. There are new parameters to the command to support file tagging. The parameters are **-T**, **-M**, and **-F**. The format of the iconv command with file tag support is:

```
iconv -T -f ISO8859-1 -t IBM-1047 file
```

-T Specifying **-T** means that the file is tagged as text; this sets the txtflag on, and the codeset will be the same as what you specified in the **-t** option. In Figure 7-4 we convert the file data which is untagged and in EBCDIC codeset to ASCII codeset, and we redirect the output to the new file asci and that file is tagged.

-F Use the input file's codeset (as defined in the file tag) as the source codeset.

-M Tag a new output file as mixed. That is, the text flag (txtflag) will be off and the value for codeset will be the same as what's specified on the **-t** option.

```
FRAMHUS@SC59:/u/framhus> iconv -T -f IBM-1047 -t ISO8859-1 data > asci
FRAMHUS@SC59:/u/framhus> ls -lT
total 24
t ISO8859-1   T=on  -rw-r--r--   1 STC      SYS1         115 Jun 13 13:53 asci
- untagged    T=off -rw-r--r--   1 STC      SYS1         115 Jun 13 13:25 data
t IBM-1047    T=on  -rw-r--r--   1 STC      SYS1         115 Jun 13 13:30 ebcd
FRAMHUS@SC59:/u/framhus>
```

Figure 7-4 Convert a file from EBCDIC to ASCII and tag it

For more information on file tagging and codeset specifications, see *z/OS UNIX System Services Planning*, GA22-7800 and *z/OS UNIX System Services Command Reference*, SA22-7802.

7.3.4 Accessing data by programs

Figure 7-5 on page 136 shows an EBCDIC program reading or writing ASCII data, and Figure 7-6 shows an ASCII-compiled program reading or writing EBCDIC data.

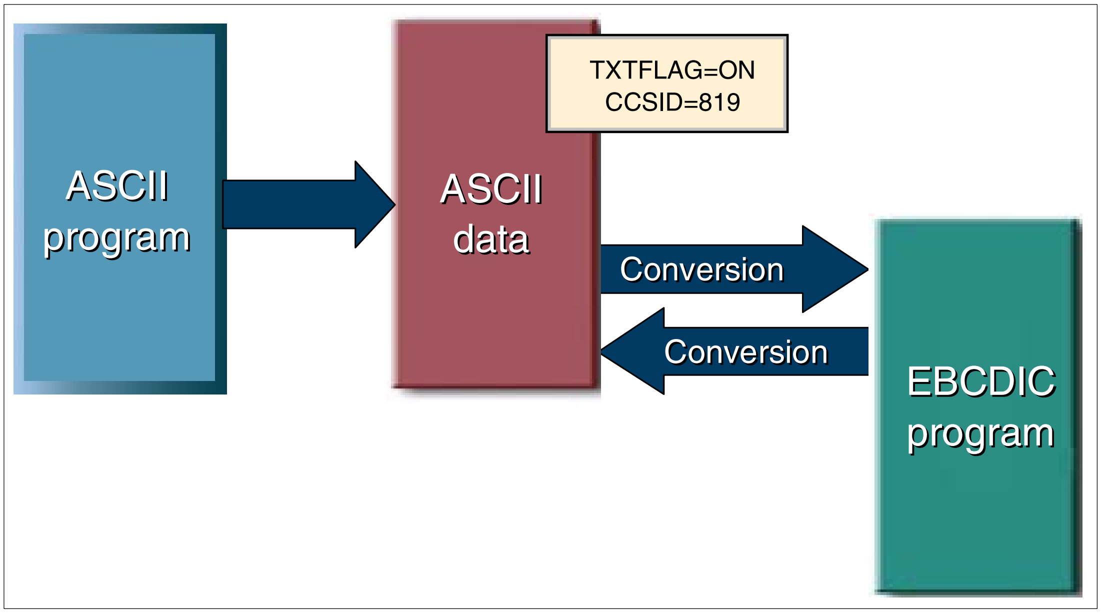


Figure 7-5 ASCII data accessed by EBCDIC program

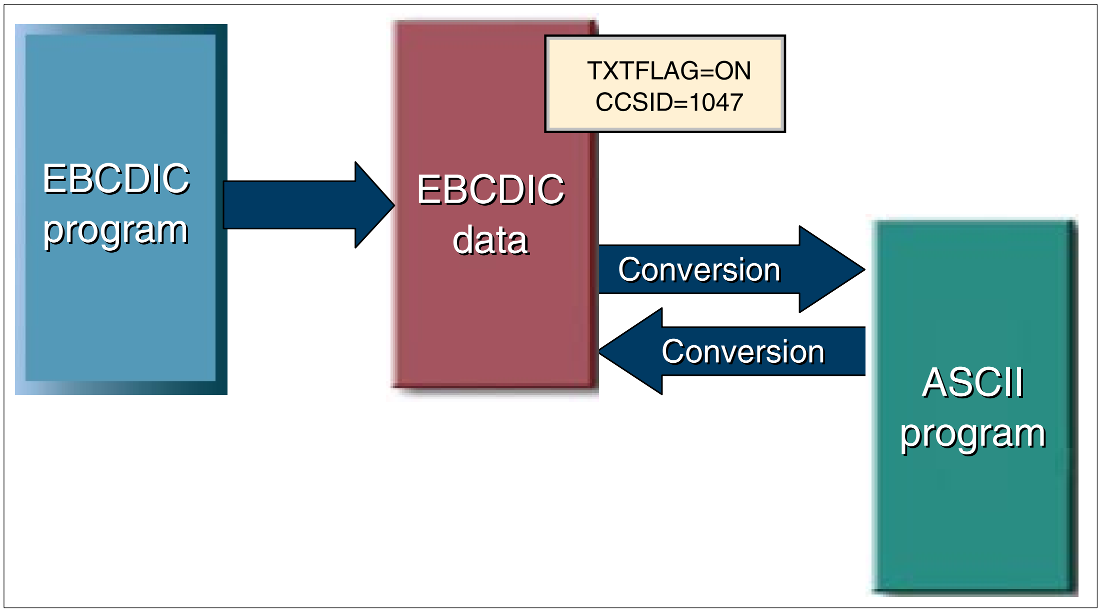


Figure 7-6 EBCDIC data accessed by ASCII program

7.3.5 Other ways to tag files

Files can also be tagged in several other ways.

Mount command

The tag parameter is new for the **mount** command. There are many ways to mount a file system:

- Parmlib statements
- Shell command
- TSO command

- ISPF shell
- BPX2MNT callable service
- REXX syscall

With all these different mount commands, as with the MOUNT statement described in “BPXPRMxx TAG parameter” on page 133, you must specify the setting of the txtflag and the codeset. Figure 7-7 shows a **mount** shell command that tags the file system as text files encoded in ASCII. Figure 7-8 shows the same function issued from REXX.

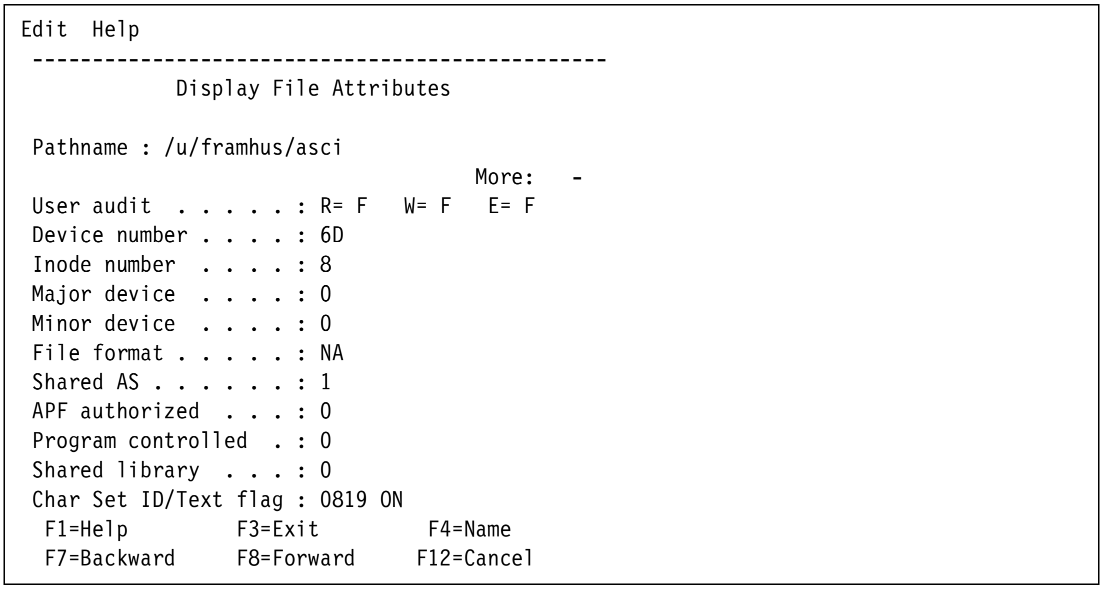
```
mount -f user.files -t hfs -c text,819 /u/lib/ascii
```

Figure 7-7 Shell mount command

```
/* rexx */
m.=’’
m.mnte_fsname=’USER.FILES’
m.mnte_path=’/u/lib/ascii’
m.mnte_type=’HFS’
m.mnte_filetag=’033380000’x /* x’0333’ = 819 */
address syscall ‘mount m.’
```

Figure 7-8 Mount command issued in REXX

There is support in the ISHELL to display the file tag. It is in the display attribute window. The CCSID is shown along with the txtflag set to ON or OFF, as shown in Figure 7-9.



```
Edit  Help
 ------------------------------------------------
             Display File Attributes

 Pathname : /u/framhus/asci
                                     More:   -
 User audit  . . . . . : R= F   W= F   E= F
 Device number . . . . : 6D
 Inode number  . . . . : 8
 Major device  . . . . : 0
 Minor device  . . . . : 0
 File format . . . . . : NA
 Shared AS . . . . . . : 1
 APF authorized  . . . : 0
 Program controlled  . : 0
 Shared library  . . . : 0
 Char Set ID/Text flag : 0819 ON
  F1=Help         F3=Exit         F4=Name
  F7=Backward     F8=Forward     F12=Cancel
```

Figure 7-9 ISHELL display file attribute

Redirection

You often use redirection to create a new file. The shell redirection defaults are no tagging, and no autoconversion.

There are two shell environment variables that can override this default, so that redirected files get tagged, as follows:

_TAG_REDIR_OUT=TXT

```
command > outfile
```

The redirected file is tagged with txtflag on and the codeset is set at the first write to outfile.

The same applies for stderr; here the shell variable is as follows:

```
_TAG_REDIR_ERR=TXT
```

Language environment and C

A new FILETAG run-time option is added for more granular control over how untagged files are set up for conversion, and whether or not open functions will tag new or empty files. The syntax of FILETAG is:

```
FILETAG (AUTOCVT | NOAUTOCVT,AUTOTAG | NOAUTOTAG)
```

The second operand activates or deactivates automatic tagging of new files when created by **fopen()**, **popen()**, or **freopen()**.

Two new C functions, **_fchattr()** and **_chattr()** are added in this release to allow you to change the attributes of a file, such as, for example, access mode and reference time. These C functions also allow you to tag the file. The provided attributes structure includes a file_tag member. When this structure is populated and passed to either function, the file_tag structure is used to immediately tag the specified file.

Automatic file tagging

When a program issues **fopen()** or **popen()** with the “text” option, and using the C-RTL FILETAG(,AUTOTAG) run-time option, any new or empty files are automatically tagged at first **write()**. Programs that use this form of opening a file are already set up for tagging, and require the least effort to set up automatic conversion.

7.3.6 C/C++

There is a new z/OS C/C++ V1R2 compiler option, ASCII, that instructs the compiler to use ISO8895-1 for its default code page rather than IBM-1047 for character constants and string literals. You can set this option if your program is processing ASCII data natively at execution time. A bit is set in the executable for use by run-time.

The NOASII compiler option, which is the default, tells the compiler to use IBM-1047 codepage.

7.3.7 Language environment runtime

A new runtime option, FILETAG is used by the application programmer to specify that HFS files are automatically converted from ASCII to EBCDIC, and whether the files will be automatically tagged with a CCSID when they are opened. This option ensures a more granular control by the application programmer over which files will be applicable for autoconversion, and whether or not new or empty HFS files will be tagged. It is assumed that the application is coded to behave upon the setting of this option.

The format of the runtime option is:

```
FILETAG=(AUTOCVT,AUTOTAG)
```

where **AUTOCVT** enables autoconversion, and **AUTOTAG** activates the automatic tagging of new or empty files during open.

7.4 Shell and utility commands

There are several enhancements to shell and utility commands.

7.4.1 Extended attribute support

The extended attribute support is enhanced with extended attribute operands on the **`test`** shell command, and the filetest built-in command in the z/OS UNIX System Services C-shell, the **`tcsh`**.

There are also enhancements to the **`find`** command, to match files with extended attributes.

test command

The **`test`** command checks for various properties of files, strings, and integers. There are now conditional tests for extended attributes of a file. The following are recognized as valid operands in the test expression:

- **`-Ea file`** True if the file has APF extended attributes.
- **`-Ep file`** True if the file has program control extended attributes.
- **`-Es file`** True if the file has shared address space extended attributes.
- **`-El file`** True if the file has shared library extended attributes (`l` is lower-case L).

There is also an enhancement to the **`test`** shell command and filetest **`tcsh`** command to support enhanced ASCII. The operators that test for file tags are:

- **`-T`** True if the file is tagged as a text file.
- **`-B`** True if the file is tagged as a binary file.
- **`file -CS codeset`** True if the file is tagged with the specified codeset.

7.4.2 Systems management commands

There are some enhancements and new parameters to the following z/OS UNIX System Services shell commands in the systems management area:

- **`df`**
- **`find`**
- **`uname`**

Display the amount of free space in the file system

The **`df`** command shows the amount of free space left in a file system. There are enhancement to the **`df`** command:

- A new **`-S`** option displays the SMF accounting fields.
- Enhancements to the **`-v`** option.

df command examples

The **`-S`** option is a new option to the **`df`** command, and displays the SMF accounting fields:

```
df -S
```

```
/Z01RA1        (HFS.ZOSR01.Z01RA1.ROOT)  152408/4096800 4294926564 Available
Number of reads                       : 18
Number of writes                      : 0
Number directory I/O blocks           : 0
Number read I/O blocks                : 0
Number write I/O blocks               : 0
Total number bytes read               : 0x0  0
Total number bytes written            : 0x0  0
/SC63          (WTSCPLX2.SC63.SYSTEM.HFS)2640/2880      4294967285 Available
Number of reads                       : 12
Number of writes                      : 0
Number directory I/O blocks           : 0
Number read I/O blocks                : 0
Number write I/O blocks               : 0
Total number bytes read               : 0x0  0
Total number bytes written            : 0x0  0
/              (WTSCPLX2.SYSPLEX.ROOT)   14128/15936    4294967198 Available
Number of reads                       : 193
Number of writes                      : 0
Number directory I/O blocks           : 37664
Number read I/O blocks                : 0
Number write I/O blocks               : 0
Total number bytes read               : 0x0  0
Total number bytes written            : 0x0  0
```

Figure 7-10 df -S command

The **-v** option to the **df** command lists more detailed information on the file system status:

```
df -v
```

```
/SC63/etc      (HFS.SC63.ETC)            73072/76320    4294967011 Available
HFS, Read/Write
File System Owner : SC63        Automove=N      Client=Y
Filetag : T=off   codeset=0
/SC63/dev      (HFS.SC63.DEV)            14080/14400    4294967280 Available
HFS, Read/Write
File System Owner : SC63        Automove=N      Client=Y
Filetag : T=off   codeset=0
/Z01RA1        (HFS.ZOSR01.Z01RA1.ROOT)  152408/4096800 4294926564 Available
HFS, Read/Write
File System Owner : SC63        Automove=Y      Client=Y
Filetag : T=off   codeset=0
/SC63          (WTSCPLX2.SC63.SYSTEM.HFS)2640/2880      4294967285 Available
HFS, Read/Write
File System Owner : SC63        Automove=N      Client=Y
Filetag : T=off   codeset=0
/              (WTSCPLX2.SYSPLEX.ROOT)   14128/15936    4294967198 Available
HFS, Read/Write
File System Owner : SC63        Automove=Y      Client=Y
Filetag : T=off   codeset=0
```

Figure 7-11 df -v command

Three new fields of information are added:

- The file system mount tag value

And, for file systems mounted in a shared HFS environment:

- The file system automove status
- The file system client status

The last two information fields are only relevant if the file system is participating in a shared HFS, otherwise they will not be displayed.

```
/              (WTSCPLX2.SYSPLEX.ROOT)   14128/15936    4294967198 Available
HFS, Read/Write
File System Owner : SC63        Automove=Y      Client=Y
Filetag : T=off   codeset=0
Number of reads                      : 246
Number of writes                     : 0
Number directory I/O blocks          : 37694
Number read I/O blocks               : 0
Number write I/O blocks              : 0
Total number bytes read              : 0x0  0
Total number bytes written           : 0x0  0
```

Figure 7-12 df -Sv command showing the sysplex root HFS

find command

The **find** command is enhanced to find files that match the extended attribute of a file, which means you can find a file meeting specified criteria.

The **find** command has a new parameter **-ext c**, that finds files that match the extended attribute specified. The extended attribute **c** can be **a**, **p**, **s**, or **l**. These extended attribute values have the following meanings:

- **a** - The program is considered loaded from an APF-authorized library.
- **p** - The program is considered program-controlled.
- **s** - The program is enabled to run in a shared address space.
- **l** - The program is a shared library object (this is a lower-case L).

Figure 7-13 shows how to find files in a directory that have the Program Control extended attribute.

```
FRAMHUS @ SC64:/u/framhus>find /bin/* -ext p
/bin/IBM/FOMOHDBX
/bin/IBM/FOMOPMD
/bin/IBM/FOMOPOE
/bin/IBM/FSUMSPSW
/bin/IBM/FSUMSSU
/bin/passwd
/bin/pdbx
/bin/pmd
/bin/poe
/bin/su
FRAMHUS @ SC64:/u/framhus>
```

Figure 7-13 Find files with an extended attribute

uname command

You can use the **uname** utility to get configuration information about the machine you are running on. The default option **-s** has previous given you OS/390 as the name of the operating system you are running on.

This utility has now been enhanced with a new parameter: **-I** (Capital letter i). This gives you the current product name of the operating system. You should use this parameter together with the **-a**, **-r**, **-s**, and **-v** options to get the correct product name, version, and release number of the operating system. If you do not specify the **-I** option, the OS/390 product name and its old anticipated release numbers are returned, as shown in Figure 7-14.

```
FRAMHUS @ SC64:/u/framhus>uname -svr
OS/390 12.00 03
FRAMHUS @ SC64:/u/framhus>
```

Figure 7-14 uname command without -I

Figure 7-15 shows the command with the **-I** option, and note the difference.

```
FRAMHUS @ SC64:/u/framhus>uname -Isvr
z/OS 02.00 01
FRAMHUS @ SC64:/u/framhus>
```

Figure 7-15 uname command with -I

7.5 Logical file system

The logical file system (LFS) component of z/OS V1R2 UNIX System Services has some additional enhancements to support new functions introduced in this release:

- TCP/IP resolver address space
- Logical file system soft shutdown
- **pread()** and **pwrite()** support

7.5.1 New TCP/IP resolver

A name resolver converts a TCP/IP hostname to an IP address, or vice versa. There are several name resolvers in TCP/IP, one in Language Environment, and one in CICS, and multiple resolver libraries exist. There are MVS native and Language Environment resolver APIs. The search order for TCPIP.DATA varies across resolver libraries. This can lead to an inconsistent name resolution process. It makes it difficult to provide resolver enhancements in a consistent and timely manner.

A new TCP/IP resolver address space supports the consolidation of the many ways to resolve host names or IP addresses. It provides for both global and local user settings to be configured.

LFS is responsible for starting the new TCP/IP resolver address space. The new address space is started during IPL. In order for LFS to start this new resolver address space, the system must be configured with an AF_INET socket file system domain name in BPXPRMxx.

There are new GetHostByName and GetHostByAddr functions in the LFS. These functions are assembler macro callable services BPX1GHN and BPX1GHA. These new callable services will now be used by TCP/IP and Language Environment. They will actually call TCP/IP code if the resolver address space is started.

BPXPRMxx parmlib member

We recommend that you update the BPXPRMxx parmlib member and use this to start the resolver address space. Using the BPXPRMxx member and OMVS initialization to start the resolver helps ensure that the resolver API is available before any /etc/rc or COMMNDxx parmlib members are executed.

The RESOLVER_PROC statement in parmlib member BPXPRMxx is used to start the resolver during z/OS UNIX System Services initialization.

The statement specifies how the resolver address space is processed during UNIX System Services initialization. The resolver address space is used by TCP/IP applications for name-to-address or address-to-name resolution.

```
RESOLVER_PROC(procname|DEFAULT|NONE)
```

The `procname` is the name of the address space for the resolver and the procedure member name in SYS1.PROCLIB.

DEFAULT causes an address space named RESOLVER to start. This also happens if the RESOLVER_PROC statement is not specified in the BPXPRMxx.

NONE specifies that no address space is to be started.

The MVS operator console command is:

```
D OMVS,OPTIONS
```

Figure 7-16 on page 144 shows the RESOLVER_PROC value used to start the resolver address space at the bottom of the display.

```
D OMVS,OPTIONS
BPXO043I 16.32.09 DISPLAY OMVS 207
OMVS     000F ACTIVE          OMVS=(2A)
CURRENT UNIX CONFIGURATION SETTINGS:
MAXPROCSYS      =        300    MAXPROCUSER     =      10125
MAXFILEPROC     =      65535    MAXFILESIZE     = NOLIMIT
MAXCPUTIME      = 2147483647    MAXUIDS         =         50
MAXPTYS         =        256
MAXMMAPAREA     =       4096    MAXASSIZE       = 2147483647
MAXTHREADS      =     100000    MAXTHREADTASKS  =      32768
MAXCORESIZE     =    4194304    MAXSHAREPAGES   =   32768000
IPCMSGQBYTES    =     262144    IPCMSGQMNUM     =      10000
IPCMSGNIDS      =      20000    IPCSEMNIDS      =      20000
IPCSEMNOPS      =      32767    IPCSEMNSEMS     =         25
IPCSHMMPAGES    =      25600    IPCSHMNIDS      =      20000
IPCSHMNSEGS     =       1000    IPCSHMSPAGES    =    2621440
SUPERUSER       = BPXROOT       FORKCOPY        = COPY
STEPLIBLIST     =
USERIDALIASTABLE=
PRIORITYPG VALUES: NONE
PRIORITYGOAL VALUES: NONE
MAXQUEUEDSIGS   =     100000    SHRLIBRGNSIZE   =   67108864
SHRLIBMAXPAGES  =       4096    VERSION         = Z02RB1
SYSCALL COUNTS  = NO            TTYGROUP        = TTY
SYSPLEX         = YES           BRLM SERVER     = SC65
LIMMSG          = NONE          AUTOCVT         = OFF
RESOLVER PROC   = DEFAULT
```

Figure 7-16 d omvs,options command display

7.5.2 Logical file system soft shutdown

A new function, soft shutdown for mounted file systems is provided through the MVS operator console Modify command:

```
F BPXOINIT,SHUTDOWN=FILESYS
```

This function is retrofitted back to OS/390 V2R7 with APAR OW48199.

Currently, it is available on the z/OS UNIX System Services home page:

```
http://www.ibm.com/servers/eserver/zseries/zos/unix
```

This function is also on the Tools and Toys page: under the heading OS/390 UNIX tools you can find a tool called BPXSTOP. This tool is used to kill processes and unmount file systems after normal shutdown procedures and termination commands for UNIX products have been issued. Not all servers and applications respond well to terminating signals. You should make every attempt to shut down all UNIX work you can, including and especially TCP/IP and NFS client and server, prior to using this function. Using BPXSTOP may not harden the data to disk when the file systems are unmounted.

Now there is a new and supported way to help terminate UNIX activity on the system. In z/OS V1R2 there is an enhancement to the existing **F BPXOINIT,SHUTDOWN=** command, which provides a method for unmounting all file systems and hardening the cached buffers to disk.

Many of the different products running under z/OS UNIX System Services, like NFS for z/OS and Communication Server, have their own shutdown procedures. These procedures should be used first to shut down the different products, before any z/OS UNIX System Services termination commands are used.

The distinction between the shared HFS file systems in a Sysplex and non-Sysplex file systems is based on whether SYSPLEX(YES) is specified in the BPXPRMxx parmlib member, and the systems are participating in a shared HFS.

If it is a non-sysplex environment, all file systems will be unmounted with the FORCE operand.

In a sysplex environment, all automounted file systems and file systems mounted on the system you shut down will be unmounted. For file systems mounted with AUTOMOVE parameter set, the ownership is moved to another system in the sysplex. The remaining file systems will be unmounted FORCE.

F BPXOINIT command

The Modify MVS console command **F BPXOINIT SHUTDOWN** is now enhanced with the FILESYS parameter:

```
F BPXOINIT SHUTDOWN=FILESYS
```

where the operand shutdown=filesys unmounts the UNIX System Services file systems as described.

7.5.3 pread() and pwrite() library function

The **pread()** z/OS C/C++ Run-time library function reads from a given position in a file, and the **pwrite()** writes from a given position in a file, without changing the file pointer(cursor).

This is part of the UNIX 98 specification. See Figure 7-17 for the prototype statement. It is a combined **lseek()** and **read()/write()** in one syscall. **lseek()** changes the file pointer to a new position in an HFS file. **read()/write()** reads/write a specified number of bytes from the current position of the file into/from a memory buffer.

This new C/C++ library function allows the manipulation of the file pointer before the I/O operation, and the file pointer does not change after the I/O operation. The difference between the normal **read()/write()** statements is an additional offset parameter. The offset indicates the adjustment in bytes to be made to the current file pointer before starting the **read()** or **write()** function.

```
ssize_t pread(int fd, void *buf, size_tt nbyte, off_ offset)
ssize_t pwrite(int fd, void *buf, size_tt nbyte, off_ offset)
```

Figure 7-17 pread() and pwrite() prototype statements

Successive executions of the statement shown in Figure 7-18 would keep on reading 20 bytes of data from the same position in the file.

```
pread(fd,buffer,20,20)
```

Figure 7-18 pread() statement

The **pread()/pwrite()** function is also implemented as an assembler callable service named BPX1RW under control of the BPXYFUIO, the User I/O block.

This function is used in a performance improvement for Lotus Domino.

7.6 HFS program control

A new RACF facility class profile is added for HFS control. The profile is called BPX.DEAMON.HFSCTL.

If you want only files loaded from HFS to be checked for program control, you can set up this facility class profile. You must have the BPX.DAEMON facility class active to use this new HFS program control.

Define the resource profile, give the users read access, and refresh the profile as shown in Figure 7-19.

```
RDEFINE FACILITY BPX.DAEMON.HFSCTL UACC(NONE)
PERMIT BPX.DAEMON.HFSCTL CLASS(FACILITY) ID(username) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

Figure 7-19   Define HFS program control

8

SMP/E enhancements

In this chapter, the following topics are discussed:

- Migrating to Z/OS Release 2 SMP/E
  - SMP/E availability
  - Improved user customization
  - Nucleus backup
- Packaging enhancements
  - 64-bit addressing
  - Longer link names
  - Conditional JCLIN
- Usability enhancements
  - SMPPTS data set management
  - HOLDDATA summary reports
- eDelivery infrastructure
  - Receive from a network

8.1 Migrating to z/OS Release 2 of SMP/E

SMP/E provides the ability to install software products and services directly from a network source, such as the Internet. Today SMP/E can only process input locally on tape or DASD. By installing directly from a network source, SMP/E enables a more seamless integration of electronic software delivery and installation. This reduces the tasks and time required to install software delivered electronically.

SMP/E also provides facilities to construct, and then later unwrap, network transportable packages of software. This allows customers and vendors to create their own packages of SMP/E installable software, and then distribute them within their own enterprise, or to other enterprises. Specifically, the packaging routine accepts partitioned or sequential data sets as input and creates a network transportable package as output. Once this package is made accessible on an FTP server, the SMP/E RECEIVE command can then be used to transfer the package through a TCP/IP network directly into a z/OS SMP/E environment.

SMP/E has several new areas of possible user customization options, with easier-to-use interfaces, and requiring no migration actions to carry forward existing customization when installing a new release of SMP/E.

- SMP/E availability
- Improved user customization
- Nucleus backup

8.2 SMP/E availability

As in previous releases of the operating system, SMP/E forms part of the entitled base elements of z/OS Release 2. There is a version change to SMP/E introduced with this new release of z/OS. SMP/E is also available to be ordered separately as an individual product.

8.3 Improved user customization

SMP/E provides several new areas for user customization. Previously, the areas that were changed had to be recreated when a new release of SMP/E is installed. The specific areas of SMP/E customization that are addressed by the current enhancements are:

- Dynamic allocation using GIMDDALC
- SMP/E exit routines

8.3.1 Dynamic allocation using GIMDDALC

In previous releases, to allocate a data set during processing, SMP/E has several sources of information to use, as follows:

- DD statement
- DDDEF entry
- GIMMPDFT module

DD statement

If a DD statement has been specified, it is used to satisfy the allocation, as shown in Figure 8-1.

```
//STEP EXEC PGM=GIMSMP
//SMPCSI  DD DSN=SMP.GLOBAL.CSI,DISP=SHR
//SMPPARM DD DSN=SMP.SMPPARM.DATA,DISP=SHR
//SMPCNTL DD *
    SET BDY(ZOSR2T).
  ...
/*
```

Figure 8-1 SMPPARM specified via a DD statement

DDDEF entry

If the SMPPARM DD statement is not present, SMP/E searches for a DDDEF entry for the ddname in the current set-to zone. If found, it is then used dynamically to allocate the data set, as shown in Figure 8-2.

```
//STEP EXEC PGM=GIMSMP
//SMPCSI  DD DSN=SMP.GLOBAL.CSI,DISP=SHR
//SMPCNTL DD *
  SET BDY(ZOSR2T).
  UCLIN.
  ADD DDDEF(SMPPARM) DATASET(SMP.SMPPARM.DATA) SHR.
  ENDUCL.
  ...
/*
```

Figure 8-2 SMPPARM specified in the DDDEF entry

GIMMPDFT module

If a DDDEF entry is not found, SMP/E looks in module GIMMPDFT for information on how to dynamically allocate the data set.

The information in GIMMPDFT applies to all zones, not just the set-to zone. Currently, module GIMMPDFT contains three sections to define allocation information for SYSOUT data sets, SMPWRK1-6 and SYSUT1-4 data sets, and SMPTLIB data sets. The default module GIMMPDFT is essentially empty and does not contain allocation information for any data sets. However, a user can change the values in GIMMPDFT with the SMP/E supplied sample USERMOD in the SYS1.SAMPLIB data set. This sample contains superzap statements for the data set entries in the three sections of GIMMPDFT.

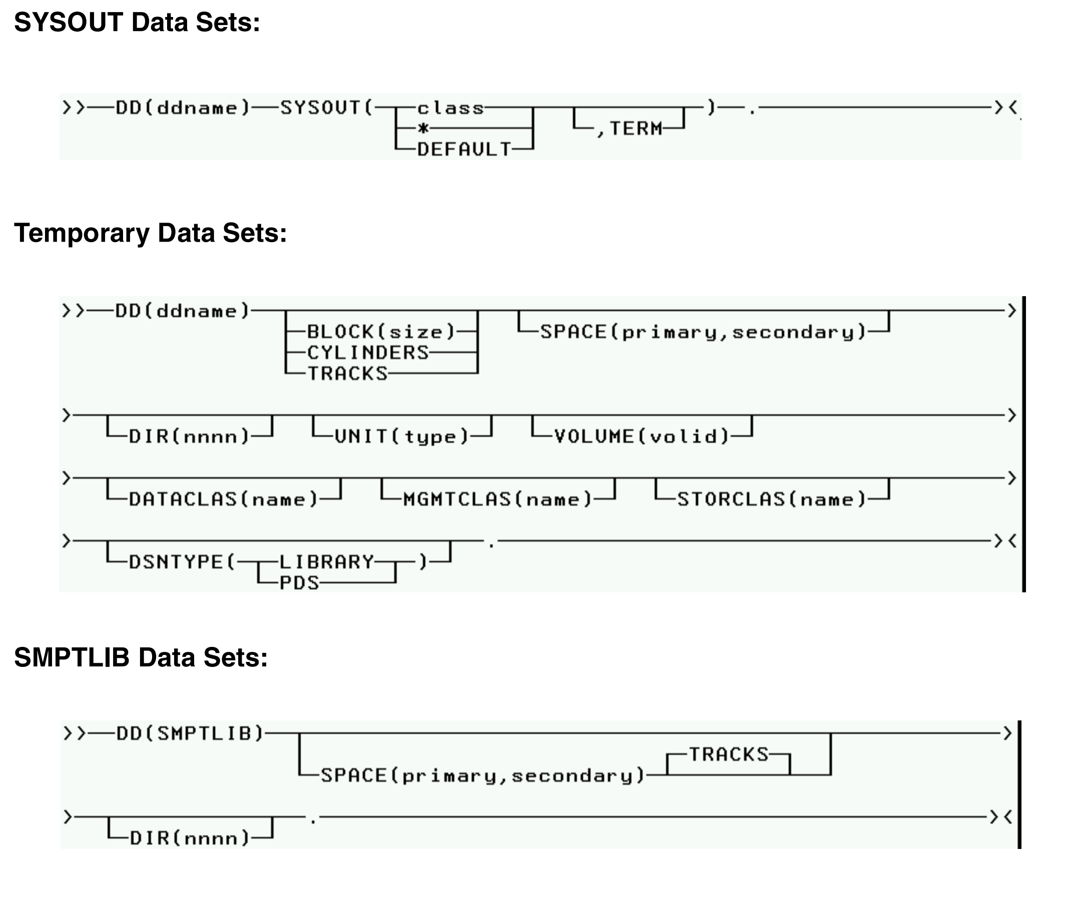
There were several problems in providing an interface such as module GIMMPDFT, as follows:

- The first is the error-prone interface a user is given to make updates to module GIMMPDFT with the use of superzap statements. A user has to translate character data to EBCDIC hex codes, determine the exact offset within the module, and then code the zap statements accordingly. This is prone to errors.
- The second is the fact that SMP/E supplies a default copy of the module GIMMPDFT. When service is applied which replaces GIMMPDFT, or when migrating to a new release of SMP/E, any changes a user may have made are overwritten and lost.

To solve these problems, a new control statement style interface has been developed called GIMDDALC.

GIMDDALC control statements

To solve this problem, SMP/E uses a new member in the existing SMPPARM data set, instead of using module GIMMPDFT to define allocations. A member, named GIMDDALC, contains text control statements to define the allocations, and SMP/E supplies a sample model member in the SYS1.SAMPLIB data set.



SYSOUT Data Sets:

```
>>──DD(ddname)──SYSOUT(─┬─class───┬──┬────────┬─)──.──────────────────><
                        ├─*───────┤  └─,TERM──┘
                        └─DEFAULT─┘
```

Temporary Data Sets:

```
>>──DD(ddname)──┬──────────────┬──┬────────────────────────────┬──────>
                ├─BLOCK(size)─┤  └─SPACE(primary,secondary)───┘
                ├─CYLINDERS───┤
                └─TRACKS──────┘

>──┬────────────┬──┬────────────┬──┬──────────────┬───────────────────>
   └─DIR(nnnn)─┘  └─UNIT(type)─┘  └─VOLUME(volid)─┘

>──┬─────────────────┬──┬─────────────────┬──┬─────────────────┬──────>
   └─DATACLAS(name)─┘  └─MGMTCLAS(name)─┘  └─STORCLAS(name)─┘

>──┬──────────────────────────┬──.────────────────────────────────────><
   └─DSNTYPE(─┬─LIBRARY─┬─)─┘
              └─PDS─────┘
```

SMPTLIB Data Sets:

```
>>──DD(SMPTLIB)──┬─────────────────────────────────────────────┬──────>
                 └─SPACE(primary,secondary)─┬─TRACKS─┬─────────┘
                                            └────────┘

>──┬────────────┬──.──────────────────────────────────────────────────><
   └─DIR(nnnn)─┘
```

Figure 8-3 GIMDDALC control statements

By copying the GIMDDALC member and placing it in the SMPPARM data set, SMP/E could then read the user-defined control statements contained within and allocate the data sets, as shown in Figure 8-4 on page 151.

Syntax rules

When creating the GIMDDALC control statement member, use the following rules:

- For SMPTLIB, only the SPACE, TRACKS and DIR operands are allowed.
- Only one GIMDDALC control statement is allowed for a single data set.
- At least one operand (other than DD) must be specified on each GIMDDALC control statement.
- GIMDDALC control statements must each start on a new line.
- GIMDDALC control statements may be continued on more than one line. SMP/E assumes a statement is continued if it does not find a period (.) before column 73.

```
/* DEFINE ALLOCATIONS FOR SYSOUT DATA SETS */
DD(SMPOUT)   SYSOUT(2,TERM)
DD(SMPLIST)  SYSOUT(2)
DD(SMPRPT)   SYSOUT(2)
DD(SMPSNAP)  SYSOUT(2)
DD(SYSPRINT) SYSOUT(*)
DD(LNKPRINT) SYSOUT(*)
/* LNKPRINT IS DEFINED IN THE LKED UTILITY ENTRY AND WILL BE USED FOR LINK-EDIT OUTPUT
*/.
DD(CPYPRINT) SYSOUT(*)
/* CPYPRINT IS DEFINED IN THE COPY UTILITY ENTRY AND WILL BE USED FOR IEBCOPY OUTPUT
*/.
DD(SYSUDUMP) SYSOUT(2).
DD(SMPPUNCH) SYSOUT(B).
DD(SMPDEBUG) SYSOUT(2).
/* DEFINE ALLOCATIONS FOR TEMPORARY DATA SETS
DD(SMPWRK1)  BLOCK(3120) SPACE(364,380) DIR(111) UNIT(SYSALLDA).
DD(SMPWRK2)  BLOCK(3120) SPACE(364,380) DIR(111) UNIT(SYSALLDA).
DD(SMPWRK3)  BLOCK(3120) SPACE(364,380) DIR(111) UNIT(SYSALLDA).
DD(SMPWRK4)  BLOCK(3120) SPACE(364,380) DIR(111) UNIT(SYSALLDA).
DD(SMPWRK6)  BLOCK(3120) SPACE(364,380) DIR(111) UNIT(SYSALLDA).
DD(SYSUT1)   BLOCK(3120) SPACE(380,760)          UNIT(SYSALLDA).
DD(SYSUT2)   BLOCK(3120) SPACE(380,760)          UNIT(SYSALLDA).
DD(SYSUT3)   BLOCK(3120) SPACE(380,760)          UNIT(SYSALLDA).
DD(SYSUT4)   TRACKS      SPACE(1,1)              UNIT(SYSALLDA).
DD(SYSPUNCH) TRACKS      SPACE(25,10)   DIR(10)  UNIT(SYSALLDA).
DD(SMPTLOAD) TRACKS      SPACE(50,20)   DIR(16)  UNIT(SYSALLDA).
/* DEFINE ALLOCATIONS FOR SMPTLIB DATA SETS.  THIS SAMPLE SMPTLIB ALLOCATION CORRESPONDS
TO SMP/E'S DEFAULT SMPTLIB ALLOCATION */
DD(SMPTLIB)  TRACKS SPACE(0,0) DIR(0).
```

Figure 8-4   GIMDDALC control statements

Syntax errors

If any syntax error is found while processing a control statement, the current record being processed is written to the SMPOUT data set, and a message is issued relevant to the error and command processing ends.

When you use this new facility when moving to a new release of SMP/E, if a user executes SMP/E using the same SMPCSI data sets with a DDDEF for SMPPARM, or using the same job stream with a DD statement for SMPPARM, then no migration actions are required in order to use existing allocation definitions.

Note: SMP/E no longer supports the GIMMPDFT module interface to define allocations. Allocations formerly defined in module GIMMPDFT must either be defined using DDDEF entries, or the new SMPPARM member GIMDDALC.

8.3.2 SMP/E exit routines

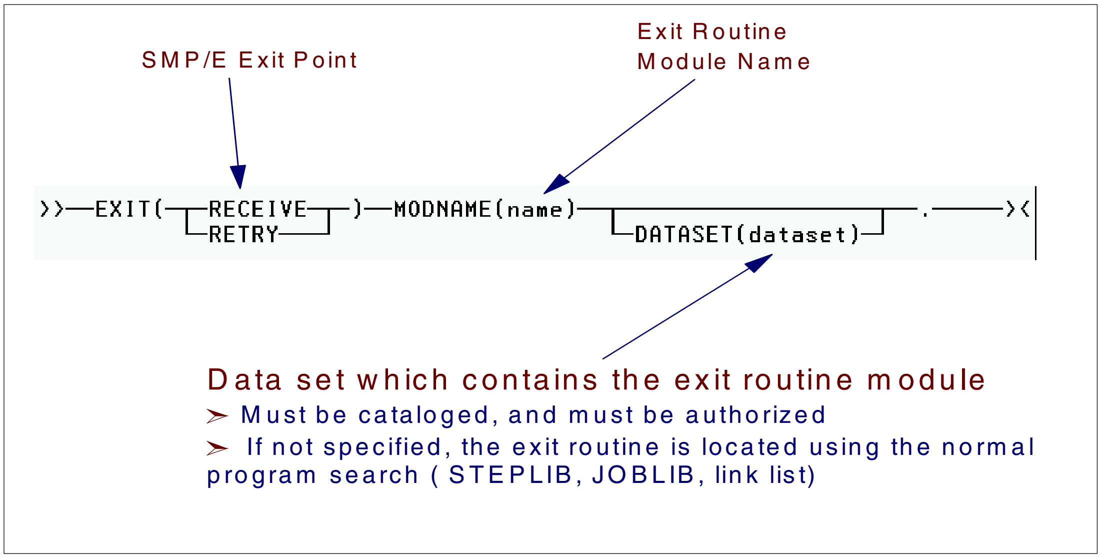
User exit routines in previous releases have been managed from an exit routine driver module named GIMMPUXD. GIMMPUXD is loaded and called to determine the actual exit routine programs which are given control at SMP/E exit points. SMP/E supplies a default GIMMPUXD module which defines no exit routines.

With this new version of SMP/E, the use of the GIMMPUXD exit routine driver is no longer supported. In its place is a new member in the SMPPARM data set, named GIMEXITS. This member contains control statements which can be used to define exit routine programs to be given control at SMP/E exit points. The control statements, shown in Figure 8-5, provide the ability to specify the identity of the data set where the exit routine resides. The exit routines should reside in any authorized data set.

Syntax rules

The following rules apply to the specification of the exit routine statement:

- Only one GIMEXITS control statement is allowed for a single exit point.
- GIMEXITS control statements must each start on a new line.
- GIMEXITS control statements may be continued on more than one line.



SMP/E Exit Point

Exit Routine Module Name

Data set which contains the exit routine module

➢ Must be cataloged, and must be authorized

➢ If not specified, the exit routine is located using the normal program search (STEPLIB, JOBLIB, link list)

Figure 8-5 GIMEXITS control statement

When processing the GIMEXITS control statements, if SMP/E determines a previous control statement for that exit point has been found, a message is issued and written to SMPOUT, and command processing ends.

A sample GIMEXITS member can be found in the SYS1.SAMPLIB data set, and is shown in Figure 8-6.

```
EXIT(RECEIVE) MODNAME(MYRECEX) DATASET(SMPE.EXITS.LOAD)

/* Defines exit routine MYRECEX in data set SMPE.EXITS.LOAD
            to get control during SMP/E RECEIVE command processing.
                                                                  */
 EXIT(RETRY)   MODNAME(MYRTYEX) DATASET(SMPE.EXITS.LOAD)

/* Defines exit routine MYRTYEX in data set SMPE.EXITS.LOAD
            to get control during SMP/E RETRY processing. */
```

Figure 8-6 GIMEXITS sample member

If a control statement has been specified for both the **RECEIVE** and **RETRY** commands, only the content of the control statement is verified.

8.4 Nucleus backup

Currently during SMP/E processing, if the IEANUC01 load module is to be updated, SMP/E optionally creates a backup copy of the module within the NUCLEUS library. The user specifies the character to be used for the name of the backup copy of IEANUC01. The backup copy is in the format of IEANUC0*x*, where *x* is the user-specified character.

Since the nucleus is now logically composed of thee physical members, SMP/E needs to extend its nucleus backup processing to all three members. This would be to manage changes to the nucleus which support the new z/Architecture hardware architecture.

However, it was found that the recovery could be partial. For some time with previous releases of MVS/ESA and OS/390, the IEANUC01 member has not contained all of the information needed for a complete rebuild, creating a potential of missing elements in the recovered system. Since IEANUC0*x* may not include everything, adding two more modules would add to the problem.

Rather than extend for the additional two members, a better solution is to remove the ability to save the nucleus. This would require that various options within SMP/E processing would need to be updated to remove the NUCID operand since this is no longer valid.

NUCID has been removed as a subentry for the OPTIONS entry, the APPLY, the LINK and RESTORE command processing. The relevant SMP/E administration dialogs that have supported the NUCID operand have also been changed to reflect the removal of the NUCID operand.

8.5 Packaging enhancements

SMP/E has several packaging enhancements being introduced with z/OS Release 2. These are:

- 64-bit addressing
- Longer link names
- Conditional JCLIN

8.5.1 64-bit addressing

The installation of software components that will be exploiting 64-bit addressing mode requires that SMP/E support two new binder options; specifically, the AMODE(64) and a new value for the COMPAT(PM4) binder option. For SMP/E to support these new binder options, SMP/E must:

- Recognize and allow software packages to specify AMODE(64) and COMPAT(PM4):

  AMODE(64) Instructs the binder to create amode 31/64 executables with 8-byte adcons.

  COMPAT(PM4) Allows the you to specify a compatibility level appropriate for amode executables.

- Save the options in the appropriate SMP/E entries.
- Pass the options to the binder when performing link edit operations.

These options may be specified on the ++MOD control statements and within JCLIN processing.

Note: This support is available on current releases via APAR IR44102

- UR52568 - OS/390 Version 2 Release 7, 8, 9, 10, and z/OS Release 1
- UR52567 - OS/390 Version 2 Release 5 and 6
- UR52566 - OS/390 Version 2 Release 4, and OS/390 Version 1 Release 3

64-bit support will not be provided for releases prior to OS/390 Version 1 Release 3.

8.5.2 Longer link names

In OS/390 Release 7, SMP/E provided support for symbolic links for elements and load modules within the hierarchical file system. The length allowed for symbolic links is set at 1023 characters. Because more products have been ported from the UNIX environment to z/OS UNIX, the 64 character limit for hard link names imposed by SMP/E has become restrictive. z/OS UNIX uses symbolic links when the 64 character limit is insufficient.

SMP/E is, therefore, increasing the length allowed for hard links to 1023 characters. This means:

- The values of the **LINK** operand on HFS element modification control statements can now be up to 1023 characters long, as shown in Figure 8-7.

```
++HFS(FILENAME) SYSLIB(SDDNAME) DISTLIB(ADDNAME)
   PARM(PATHMODE(7,5,5))
   LINK('../shortone',
        '../this/name/is/greater/than/64/characters/long/I/wonder/
 how/anyone/remembers/this/very/long/name')
   SYMLINK('../alternate/name')
   SYMPATH('../shortone').
```

Figure 8-7 Long link name

- *Alias* values on link-edit ALIAS control statements can now be up to 1023 characters long, as shown in Figure 8-8 on page 154. This requires that the alias value be enclosed in apostrophes only if a (“,”) or an apostrophe (“ ‘ ”) is contained in the value, or it spans more that one line.

```
++JCLIN.
 //LINK      EXEC PGM=IEWL,PARM='CASE(MIXED)'
 //SYSLMOD   DD PATH='/path/'
 //*LIBRARYDD=SDDNAME
 //SYSLIN    DD *
   INCLUDE DISTLIB(MODNAME)
   ALIAS '../shortone'
   ALIAS '../this/name/is/greater/than/64/characters/long/I/wonder/+
  how/anyone/remembers/this/very/long/name'
   ALIAS (SYMLINK,'../alternate/name')
   ALIAS (SYMPATH,'../shortone')
   NAME LMODNAME  RC=0
```

Figure 8-8 Link-edit alias name

8.5.3 Conditional JCLIN

The objective of conditional JCLIN processing is to allow the ability to provide multiple structural definitions within a single JCLIN input stream, such that SMP/E can pick parts of the JCLIN input stream to process based on control statements placed in the stream. The function allows the packager to use special JCL comments in the JCLIN input, which can cause SMP/E to skip over parts of the JCLIN input. The parts of the JCLIN input that are skipped are not processed by JCLIN processing and do not contribute to the structure information derived via JCLIN processing.

SMP/E control statements in JCLIN input

The conditional JCLIN processing introduces three control statements that are coded as JCL comments, but that have special processing associated with them for SMP/E JCLIN processing. Each control statement starts with "**SMP/E-**" in order to ensure that SMP/E is not inadvertently processing a real comment as a control statement.

The new JCLIN control statements are as follows:

- **//*SMP/E-IF SYSMOD(***sysmod-id***) THEN DO**

  The **SMP/E-IF** control statement specifies a *sysmod-id* that is to have checks performed against it and a **DO** that starts a **DO/END** grouping of records within the JCLIN input stream. When SMP/E JCLIN processing encounters the SMP/E control statement, the syntax is checked and the *sysmod-id* value is saved for possible checking. SMP/E also notes that a **DO** group has been initiated so that it can match the current **DO** to its **SMPE-END** control statement.

  If the SYSMOD specified on the **SMPE-IF** statement is present in the current zone (installed, superseded, or currently being installed), then the JCLIN following the **SMP-IF** statement is processed.

  If the SYSMOD is not present in the current zone (deleted, not installed, and not currently being installed), then the JCLIN up to the matching **SMP-END** statement, SMP/E issues an error message and terminates JCLIN processing.

- **//*SMPE-ELSE DO**

  The **SMPE-ELSE** clause of an **SMPE-IF** statement is optional. It is used to group together a portion of the JCLIN input stream that is to be fully processed only if the checks against the *sysmod-id* specified on the associated **SMPE-IF** statement causes the **DO/END** group of the **SMPE-IF** to be skipped. Conversely, the **DO/END** of the **SMPE-ELSE** clause is skipped if the **DO/END** group of the associated **SMPE-IF** clause is fully processed.

- **//*SMPE-END**

  The **SMPE-END** statement ends a **DO** group started on either an **SMPE-IF** statement or an **SMPE-ELSE** statement. The **SMP-END** statement is not optional: it must be used to end the **DO** group.

These conditional JCLIN statements are shown in an example in Figure 8-9 on page 156.

```
++JCLIN.
 //*SMPE-IF SYSMOD(JBB9999) THEN DO
 //STEP1   EXEC PGM=IEWL,PARM='RENT,REUS'
 //*SMPE-END
 //*SMPE-ELSE DO
 //STEP1   EXEC PGM=IEWL,PARM='RENT'
 //*SMPE-END
 ...
 ...
 //*SMPE-IF SYSMOD(JBB9999) THEN DO
 //SYSLIN  DD *
   INCLUDE AOS99(MAYBEMOD)
   INCLUDE AOS99(OKMOD)
   ...
 //*SMPE-END
 //*SMPE-ELSE DO
 //SYSLIN  DD *
   INCLUDE AOS99(OKMOD)
   ...
 //*SMPE-END
 /*
```

Figure 8-9 Conditional JCLIN example

8.6 Usability enhancements

SMP/E has several usability enhancements being introduced with z/OS Release 2. These are:

- SMPPTS data set management
- HOLDDATA summary reports

8.6.1 SMPPTS data set management

Users currently use different techniques to maintain the SMPPTS data set below a one volume limit. The SMPPTS data set is used by SMP/E to stage PTFs before they are to be installed. With the volume of SYSMODs being processed by a z/OS system, this has resulted in many cases where the size of the SMPPTS may exceed one volume. Since the SMPPTS data set is defined as a PDS, it cannot be larger than one volume.

SMP/E spill data sets

To accommodate the ever growing number of SYSMODs in an z/OS environment, SMP/E processing has been changed to allow spill data sets to be defined to contain the overflow when the SMPPTS data set is full. All spill data sets have attributes similar to the primary SMPPTS data set. Figure 8-10 on page 157 is a graphical representation of spill data set usage.

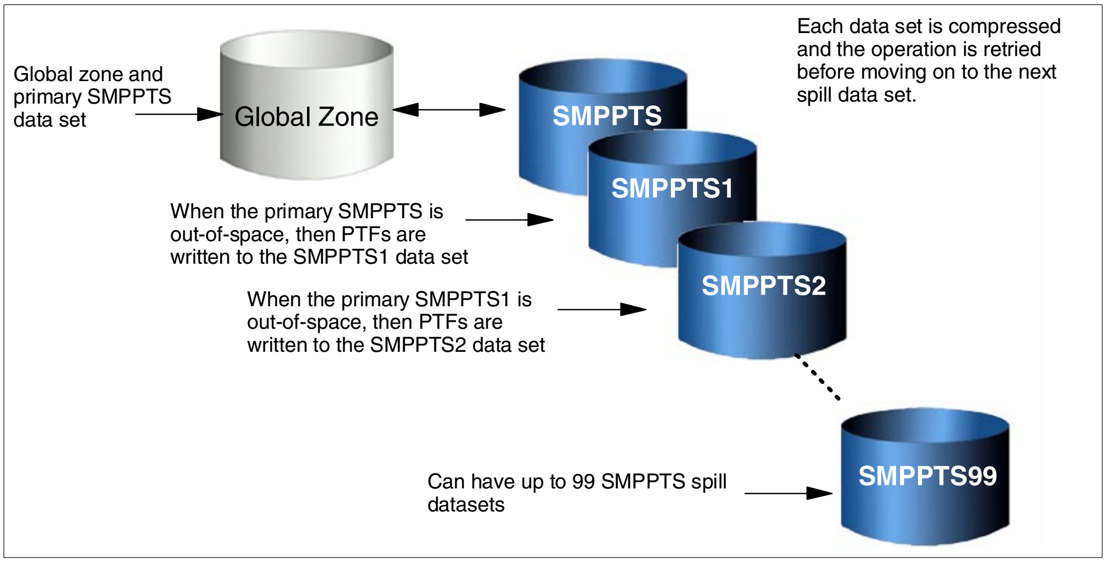

Each data set is compressed and the operation is retried before moving on to the next spill data set.
Global zone and primary SMPPTS data set
Global Zone
SMPPTS
SMPPTS1
When the primary SMPPTS is out-of-space, then PTFs are written to the SMPPTS1 data set
SMPPTS2
When the primary SMPPTS1 is out-of-space, then PTFs are written to the SMPPTS2 data set
Can have up to 99 SMPPTS spill datasets
SMPPTS99

Figure 8-10 Spill data sets overview

Using SMPPTS spill data sets

To install and customize this new support for spill data sets, do the following:

- Allocate permanent data sets for SMPPTS spill data sets.
- When accessing the logical SMPPTS data sets, SMP/E looks for reserved ddnames of SMPPTS, SMPPTS1, SMPPTS2, up to SMPPTS99.
- Specify the SMPPTS spill data sets using one of the following:
  - DD statements in the job, as shown in Figure 8-11.
  - DDDEF entries in the zones.
    The use of DDDEFs requires that *all* spill data sets be defined in *all* zones.
  - Must specify all spill data sets to be used in sequence:
    SMPPTS1 - SMPPTSn, without skipping SMPPTSn-1.
  - If SMPPTSn data sets are allocated (via DD or DDDEF), then SMP/E will use them as necessary to store, locate, and delete members.

```
//step    EXEC PGM=GIMSMP
//SMPCSI  DD DSN=SMP.GLOBAL.CSI,DISP=SHR
//SMPPTS  DD DSN=SMP.SMPPTS.DATA,DISP=SHR
//SMPPTS1 DD DSN=SMP.SMPPTS1.DATA,DISP=SHR
//SMPPTS2 DD DSN=SMP.SMPPTS2.DATA,DISP=SHR
...
/*
```

Figure 8-11 JCL spill data set DDs

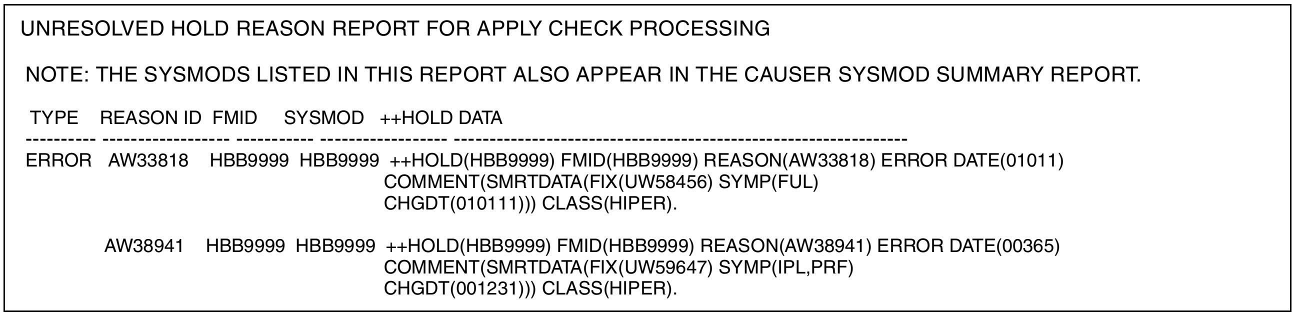
8.6.2 HOLDDATA summary reports

Currently, HOLDDATA is printed in SMP/E reports in a different location than the error messages, and informational messages which refer to the HOLDDATA. Consolidating the HOLDDATA information to be displayed immediately after the message which refers to the HOLDDATA saves the user in research time.

There are three reports provided for APPLY and ACCEPT processing that summarize the HOLD information encountered for the set of SYSMODs being installed, as follows:

- Unresolved HOLDs
- Bypassed HOLDs
- Summary HOLDs.

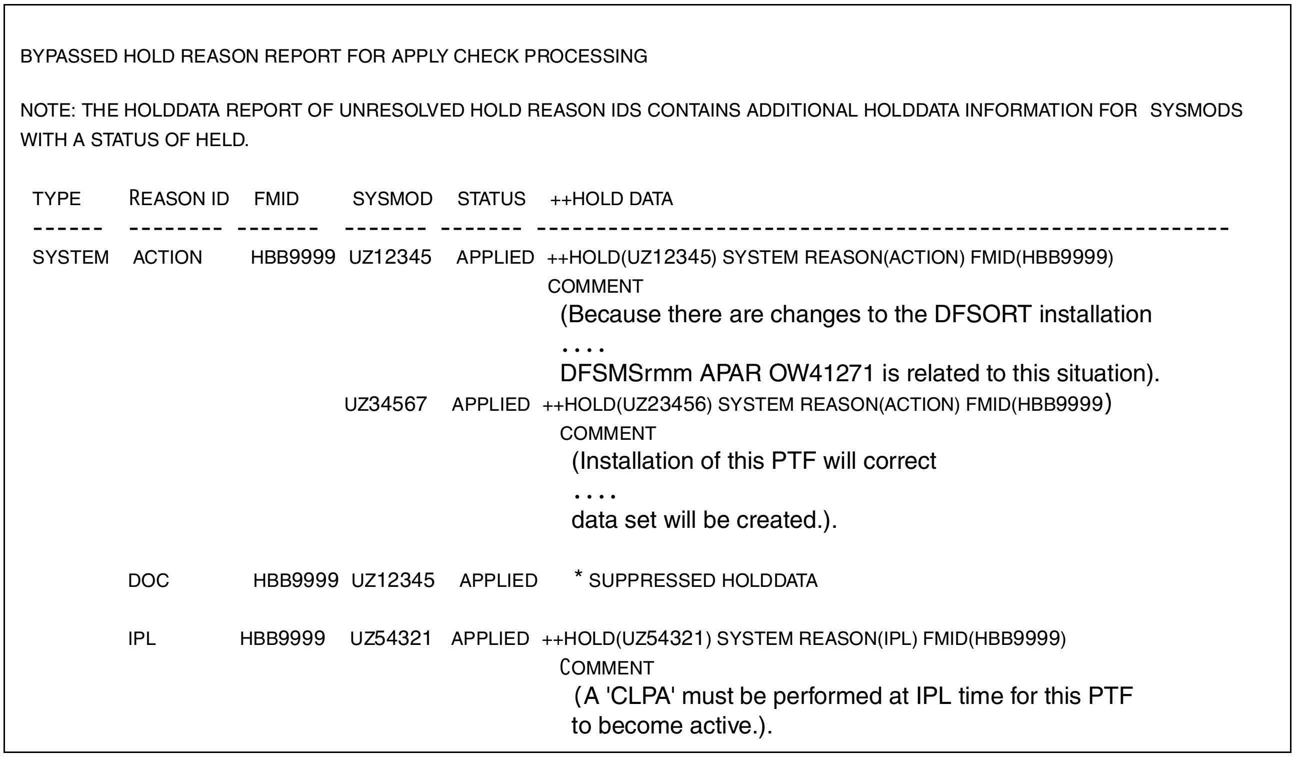
Figure 8-12 displays the Unresolved HOLDs which are the PTFs that fail, usually with ERROR holds.



```
UNRESOLVED HOLD REASON REPORT FOR APPLY CHECK PROCESSING

 NOTE: THE SYSMODS LISTED IN THIS REPORT ALSO APPEAR IN THE CAUSER SYSMOD SUMMARY REPORT.

  TYPE    REASON ID  FMID     SYSMOD   ++HOLD DATA
 ---------- ------------------ ----------- ------------------ ------------------------------------------------------------------
 ERROR   AW33818    HBB9999  HBB9999  ++HOLD(HBB9999) FMID(HBB9999) REASON(AW33818) ERROR DATE(01011)
                                      COMMENT(SMRTDATA(FIX(UW58456) SYMP(FUL)
                                      CHGDT(010111))) CLASS(HIPER).

         AW38941    HBB9999  HBB9999  ++HOLD(HBB9999) FMID(HBB9999) REASON(AW38941) ERROR DATE(00365)
                                      COMMENT(SMRTDATA(FIX(UW59647) SYMP(IPL,PRF)
                                      CHGDT(001231))) CLASS(HIPER).
```

Figure 8-12 Report of unresolved HOLDs

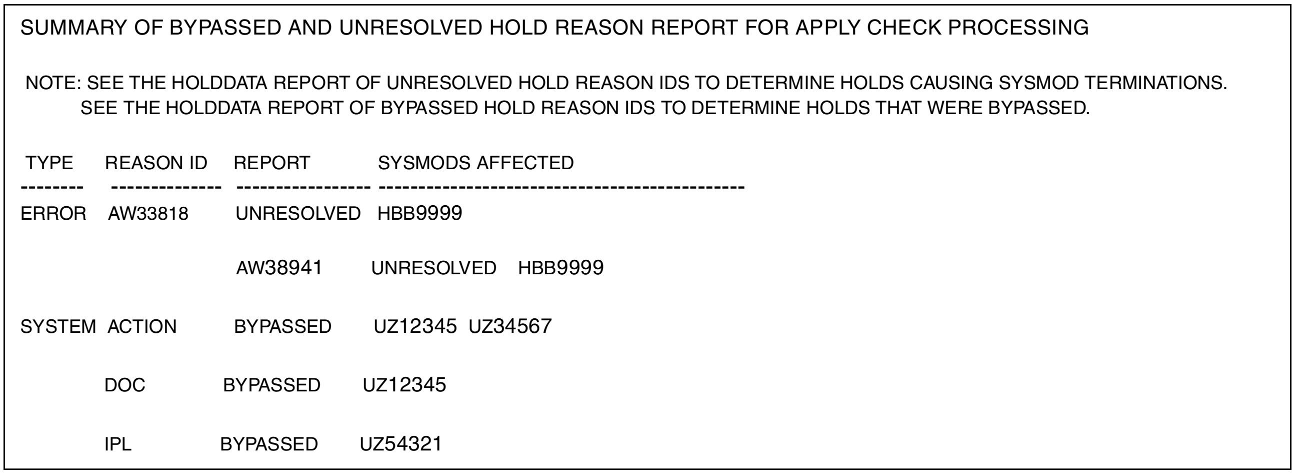
Figure 8-13 displays the Bypassed HOLDs, which are usually for PTFs with SYSTEM holds.



```
BYPASSED HOLD REASON REPORT FOR APPLY CHECK PROCESSING

NOTE: THE HOLDDATA REPORT OF UNRESOLVED HOLD REASON IDS CONTAINS ADDITIONAL HOLDDATA INFORMATION FOR  SYSMODS
WITH A STATUS OF HELD.

 TYPE      REASON ID   FMID       SYSMOD   STATUS    ++HOLD DATA
 ------   --------  -------   -------  -------  ----------------------------------------------------------
 SYSTEM    ACTION      HBB9999  UZ12345    APPLIED  ++HOLD(UZ12345) SYSTEM REASON(ACTION) FMID(HBB9999)
                                                    COMMENT
                                                     (Because there are changes to the DFSORT installation
                                                     . . . .
                                                     DFSMSrmm APAR OW41271 is related to this situation).
                                    UZ34567    APPLIED  ++HOLD(UZ23456) SYSTEM REASON(ACTION) FMID(HBB9999)
                                                     COMMENT
                                                      (Installation of this PTF will correct
                                                      . . . .
                                                      data set will be created.).

           DOC          HBB9999  UZ12345    APPLIED     * SUPPRESSED HOLDDATA

           IPL         HBB9999    UZ54321   APPLIED  ++HOLD(UZ54321) SYSTEM REASON(IPL) FMID(HBB9999)
                                                     COMMENT
                                                      (A 'CLPA' must be performed at IPL time for this PTF
                                                      to become active.).
```

Figure 8-13 Report of bypassed Holds

Figure 8-14 on page 159 displays the Summary report of HOLDs; these are usually for PTFs with SYSTEM holds.



```
SUMMARY OF BYPASSED AND UNRESOLVED HOLD REASON REPORT FOR APPLY CHECK PROCESSING

 NOTE: SEE THE HOLDDATA REPORT OF UNRESOLVED HOLD REASON IDS TO DETERMINE HOLDS CAUSING SYSMOD TERMINATIONS.
       SEE THE HOLDDATA REPORT OF BYPASSED HOLD REASON IDS TO DETERMINE HOLDS THAT WERE BYPASSED.

 TYPE    REASON ID   REPORT        SYSMODS AFFECTED
--------  ----------  ------------ ----------------------------------
ERROR   AW33818     UNRESOLVED  HBB9999

                    AW38941      UNRESOLVED   HBB9999

SYSTEM  ACTION      BYPASSED     UZ12345  UZ34567

        DOC        BYPASSED     UZ12345

        IPL        BYPASSED     UZ54321
```

Figure 8-14 Summary report of HOLDs

8.7 eDelivery infrastructure

eDelivery infrastructure provides a facility to enable SMP/E to RECEIVE input from a network location—specifically, to RECEIVE software packages over a network rather than only from tape or DASD. This enables a seamless integration of internet and SMP/E installation. This facility can also be used within an intranet environment. There are two distinct components, as shown in Figure 8-15 on page 160, and they provide the following functions:

- GIMZIP network packaging service routine, which creates transportable packages of:
  - Modification control statements (MCS)
  - RELFILEs
  - HOLDDATA
  - ANY other associated data (doc, samples, etc.)
- Network RECEIVE function, which:
  - Physically transfers the packages across a TCP/IP network
  - Extracts the original data from the packages
  - Performs traditional RECEIVE operations on the original data
  - GIMUNZIP service routine to extract the original data from the packages

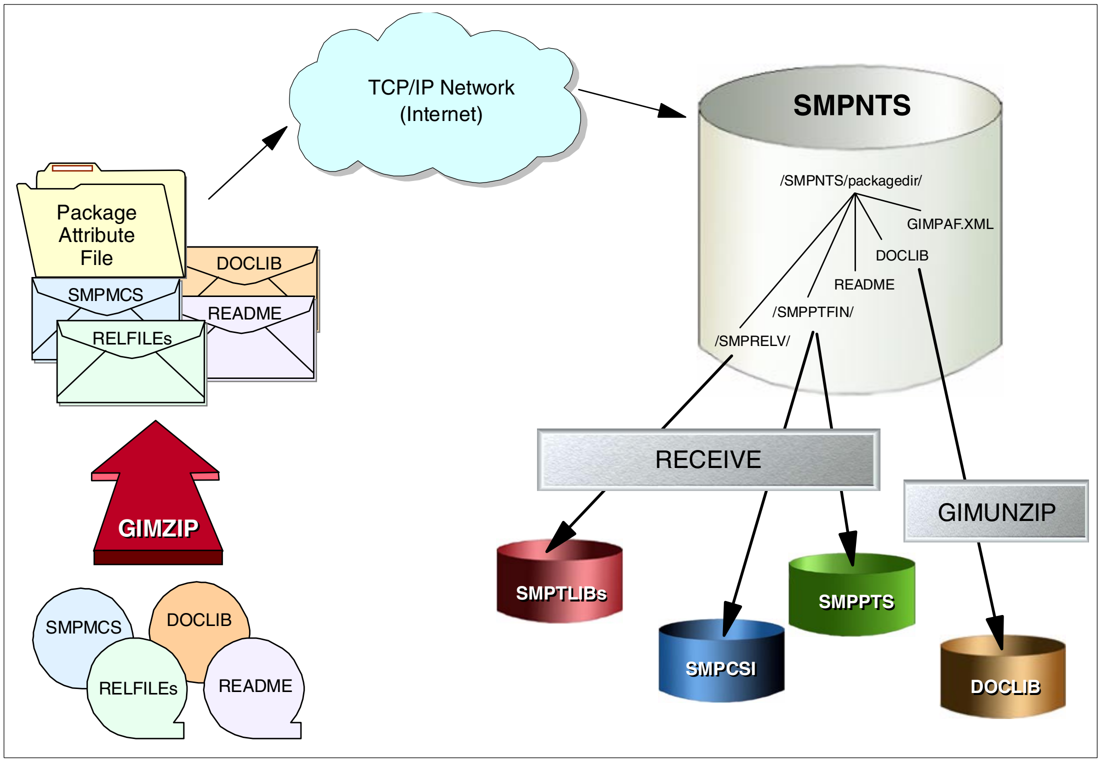


Figure 8-15 RECEIVE from a network

8.7.1 GIMZIP network packaging service routine

The GIMZIP function is used to create network transportable packages that consist of archives. The packages are then used as input to the RECEIVE command processing, which then stores them in the SMPNTS directory

Figure 8-16 shows the package control statements required for the GIMZIP function.

```
GIMZIP
   description="package description">
   <FILEDEF
     name="data set name"
     volume="volser"
     type="SMPPTFIN | SMPHOLD | SMPRELF | README"
     description="file description" />
 </GIMZIP>
```

Figure 8-16 GIMZIP input

The `FILEDEF` statements define the input data sets for GIMZIP:

Name Identifies the data set name

Volume Identifies the VOLSER if the data set is not cataloged

Type Identifies the type of data within the data set

Description Provides a text description for the data set

Each **FILEDEF** defines a single input data set which becomes a single transportable image, or archive. When you specify **type= README**, this indicates the data is intended to be viewable, and therefore is not to be archived.

GIMZIP can process many data sets at once to create a single logical package.

Calling GIMZIP

GIMZIP is a separate load module residing in the SYS1.MIGLIB library. Figure 8-17 shows the JCL statements and package control statements needed to call the GIMZIP program. The GIMZIP program is used to create a network transportable package to be placed in the directory identified by the SMPDIR DD statement.

```
//ZIPSTEP  EXEC PGM=GIMZIP
//SMPOUT   DD SYSOUT=*
//SYSUT2   DD UNIT=SYSALLDA,SPACE=(CYL,(200,20))
//SYSUT3   DD UNIT=SYSALLDA,SPACE=(CYL,(50,10))
//SYSUT4   DD UNIT=SYSALLDA,SPACE=(CYL,(25,5))
//SMPDIR   DD PATH='/package directory/'
//SYSIN    DD *
<GIMZIP
  description="Package for FMID HMP1D00">
  <FILEDEF
    name="USER.HMP1D00.SMPMCS"
    type="SMPPTFIN" />
  <FILEDEF
    name="USER.IBM.HMP1D00.F1"
    type="SMPRELF" />
  <FILEDEF
    name="USER.IBM.HMP1D00.F2"
    type="SMPRELF" />
  <FILEDEF
    name="USER.HMP1D00.README.TXT"
    type="README" />
  <FILEDEF
    description="Documents including PGM DIR"
    name="USER.HMP1D00.DOCLIB" />
</GIMZIP>
/*
```

Figure 8-17 GIMZIP usage JCL

A GIMZIP archive is composed of two files:

- The input data set named on the FILEDEF statement.
- A File Attribute File (FAF) to describe the data set.

  The FAF contains all the information needed to reconstruct the original data set, as follows:

  - The data set name, and type attribute if specified (SMPPTFIN, SMPHOLD, SMPRELF, or README)
  - Dsntype (LIBRARY[PDSE] or PDS), dsorg (PO or PS), recfm, and lrecl
  - Blksize, space, and directory blocks

The archive for the two files is created using **pax** with the compress option (**-z**)

- The two files are first copied into a work directory in the HFS so **pax** can process them

The resultant archive file resides in the package directory in the HFS, as shown in Figure 8-18.

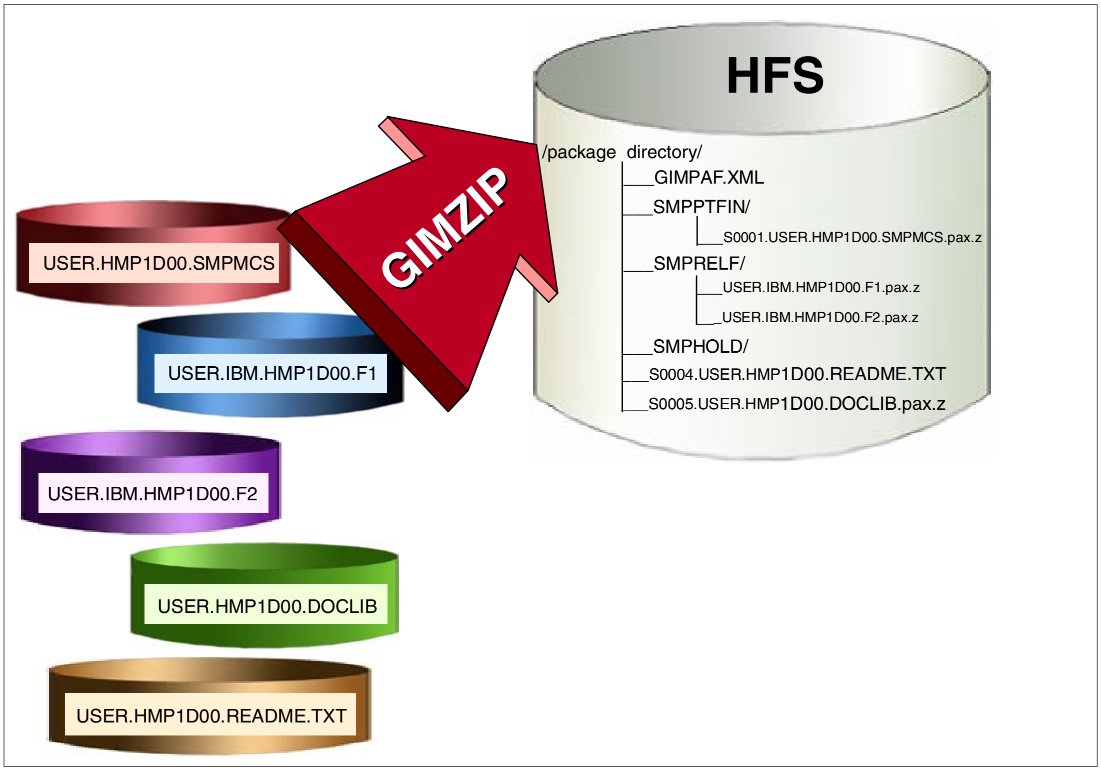


Figure 8-18 GIMZIP package contents

GIMZIP package attribute file

The package attribute file (PAF) contains the package attribute statements that describe the contents of the package and information about how the package was created. The format of the statements is shown in Figure 8-19.

```
<PKGDEF
   description="package description"
   level="vv.rr.mm.pp"
   date="yyyy.ddd"  gmt="hh:mm:ss"
   files="number of package files"
   size="package size"  originalsize="sum data set size">
   <ARCHDEF
     name="archive file name"
     type="SMPPTFIN | SMPHOLD | SMPRELF | README"
     description="archive description"
     size="archive size"  originalsize="data set size"
     hash="archive hash value" />
</PKGDEF>
<?PKGHASH hash="package hash value" ?>
```

Figure 8-19 PAF control statements

The GIMZIP package attribute file describes a transportable software package. The ARCHDEF statements each identify a single archive file associated with the package, as follows:

- Name identifies the file name of the archive.

- Type identifies the type of data within the archive.
- Description provides a description for the archive.
- Size indicates the size (in bytes) for the archive.
- Hash identifies the SHA-1 hash value (in hex) for the archive file. The hash attribute on the PKGHASH instruction identifies the SHA-1 hash value (in hex) for the PAF.

Note: Cryptographic services (ICSF) is used to compute SHA-1 hash values.

The file name of the PAF is GIMPAF.XML; it resides in the package directory within the HFS.

8.7.2 Network RECEIVE function

The first SMP/E command to process SYSMODs and HOLDDATA is the RECEIVE. The process reads data from the tape files or DASD data sets into the global zone for later SMP/E processing. Figure 8-20 on page 163 illustrates the Network RECEIVE function.

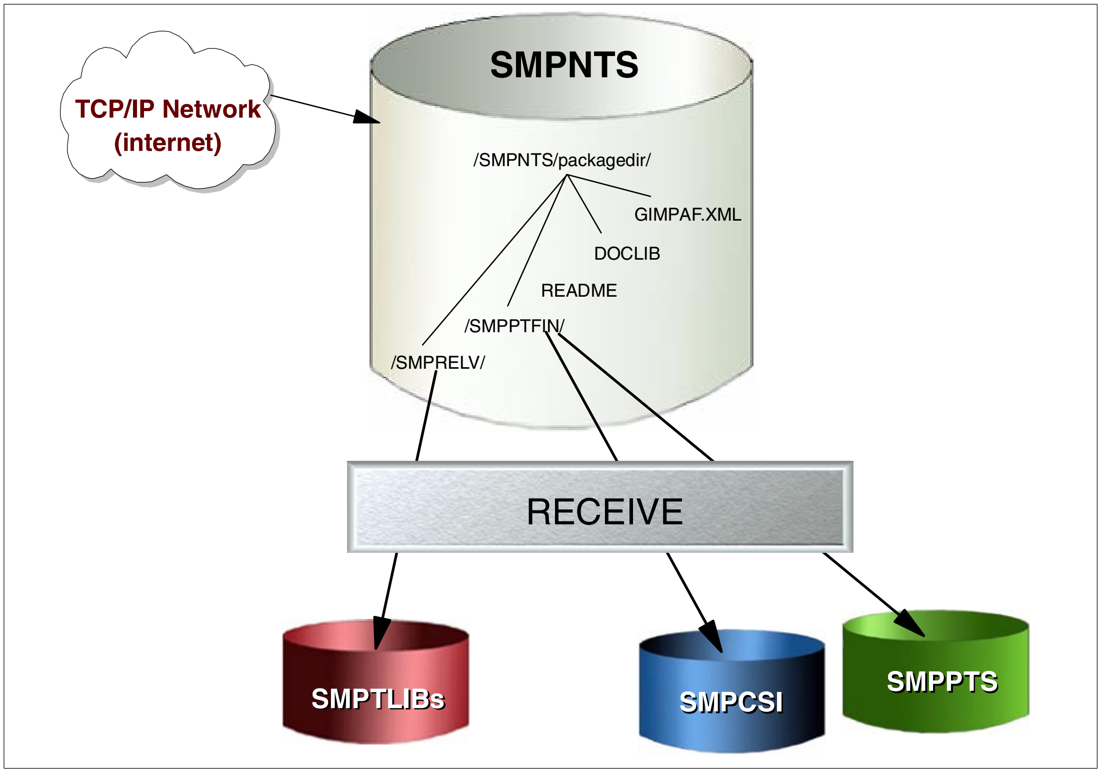


Figure 8-20 Network RECEIVE function

Network-enabled installation allows SMP/E to use data on an IP-connected server as input to the RECEIVE, and alternatively process the input retrieved earlier and store it in the SMPNTS directory of the HFS. Figure 8-21 on page 164 displays the updated receive command syntax.

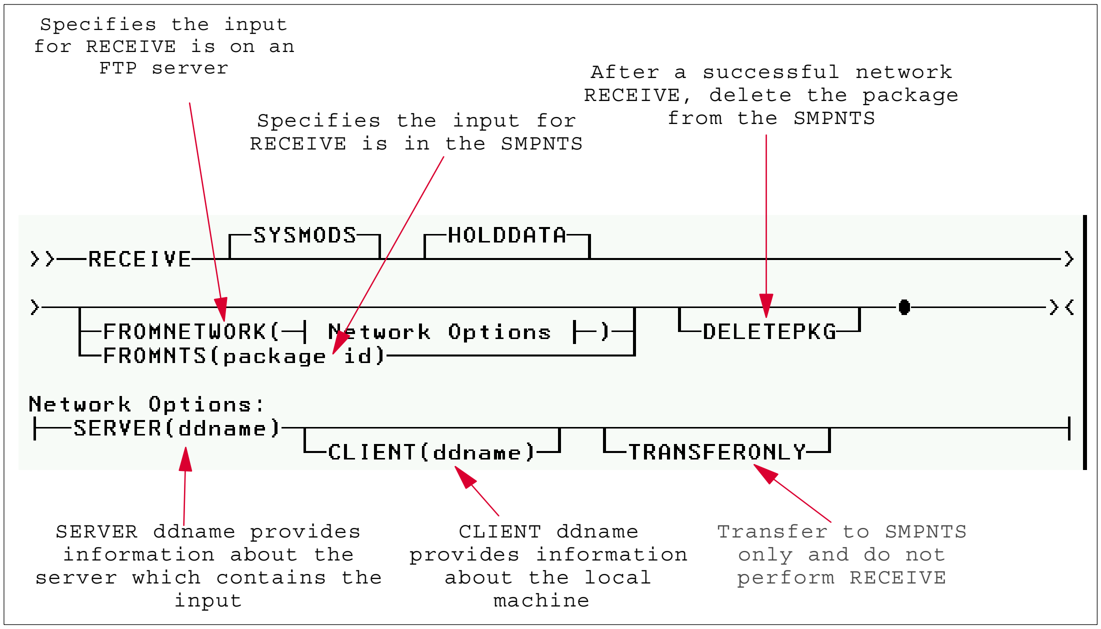

Specifies the input for RECEIVE is on an FTP server
After a successful network RECEIVE, delete the package from the SMPNTS
Specifies the input for RECEIVE is in the SMPNTS
>>—RECEIVE
SYSMODS
HOLDDATA
FROMNETWORK(
Network Options
)
FROMNTS(package id)
DELETEPKG
Network Options:
SERVER(ddname)
CLIENT(ddname)
TRANSFERONLY
SERVER ddname provides information about the server which contains the input
CLIENT ddname provides information about the local machine
Transfer to SMPNTS only and do not perform RECEIVE

Figure 8-21 Updated RECEIVE command syntax

The new operands on the RECEIVE command are as follows:

- SERVER data sets
- CLIENT data set
- FROMNETWORK processing
- FROMNTS processing

These data sets and processing options must be defined.

RECEIVE SERVER data set

The SERVER data set defines the input for a network RECEIVE operation, as shown in Figure 8-22.

```
<SERVER
   host="host name | host ip address"
   port="port number"
   user="userid"
   pw="password"
   account="account information" >
   <PACKAGE
     file="package attribute file name"
     hash="package hash value"
     id="package identifier" />
</SERVER>
```

Figure 8-22 Defining the SERVER data set

The SERVER data set contains information about:

- TCP/IP-connected host running an FTP server
- A transportable software package on that host:

  file Identifies the package attribute file for the package

  hash Identifies the SHA-1 hash value (in hex) for the PAF

id Specifies a package identifier that becomes the package directory in the SMPNTS

RECEIVE CLIENT data set

```
<CLIENT
   pasv="yes"
   retry="n" >
   <FIREWALL>
     <SERVER
       host="host name | host ip address"
       port="port number"
       user="userid"
       pw="password"
       account="account information" />
     <FIRECMD> firewall specific commands </FIRECMD>
   </FIREWALL>
 </CLIENT>
```

Figure 8-23 Defining the CLIENT data set

The CLIENT data set contains information about the TCP/IP environment of the local client machine.

This may include information required to penetrate the local firewall. FIRECMD may use substitution variables for values of remote host, user, password, etc. from the SERVER data set.

RECEIVE FROMNETWORK processing

This processing does a physical transfer of a package from a TCP/IP-connected server to the client machine, and it works in the following way:

- The SERVER data set identifies the PAF for the package, therefore it is retrieved.
- The PAF describes the archives of the package, therefore they are each retrieved.
- The PAF and archives for the package are stored in a subdirectory of the SMPNTS.

  The package identifier (in the SERVER data set) is the subdirectory name.
- The SHA-1 hash values are used to determine the integrity of the PAF and the archive files.

  If necessary, SMP/E will retry the network operation.

  Note: Cryptographic Services (ICSF) is used to compute SHA-1 hash values
- For restart ability, SMP/E checks the SMPNTS for what has already been transferred. The hash value is used to determine correctness of any existing archives

RECEIVE FROMNTS processing

The RECEIVE FROMNTS processing provides the means to:

- Extract the original data from the package as follows:
  - The package resides in a subdirectory of the SMPNTS.
  - Use **pax** to uncompress and expand the following archives: SMPPTFIN, SMPHOLD, and SMPRELF.
  - Reconstruct the original data using information from the file attribute file in the archive.

- Perform traditional RECEIVE operations on the original data:
  - Copy RELFILEs to SMPTLIB data sets.
  - Update global zone and SMPPTS from SMPPTFIN and SMPHOLD.

8.7.3 GIMUNZIP service routine

The GIMUNZIP is used to extract the original data from those archives created by the GIMZIP program. This program resides in the MIGLIB library. Figure 8-24 displays the GIMUNZIP control statements.

```
<GIMUNZIP>
   <ARCHDEF
     name="archive file name"
     volume="volser"
     prefix="data set prefix"
     newname="data set name" />
 </GIMUNZIP>
```

Figure 8-24 GIMUNZIP statements

The ARCHDEF statements define the input archives for GIMUNZIP as follows:

name Identifies the archive file.

volume Identifies the VOLSER to use for the output data set.

prefix Identifies the data set prefix for the output data set.

newname Provides a new name for the output data set

Each ARCHDEF defines a single archive file to be expanded into its original data set as follows:

- Use `pax` to uncompress and expand the archive.
- Reconstruct the original data using the information from the file attribute file in the archive.

Figure 8-25 shows the JCL statements needed to call GIMUNZIP. The user can use the GIMUNZIP program to extract files from a network transported package. In Figure 8-25, the HFS directory that is to be used is specified on the SMPDIR DD statement. The package control statements are specified on the SYSIN DD statement. The GIMUNZIP program extracts the files from the archives identified in the SYSIN DD statement.

```
//UNZIP    EXEC PGM=GIMUNZIP,PARM="HASH=YES"
//SMPOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUT3   DD UNIT=SYSALLDA,SPACE=(CYL,(50,10))
//SYSUT4   DD UNIT=SYSALLDA,SPACE=(CYL,(25,5))
//SMPDIR   DD PATH='/u/smpe/SMPNTS/HMP1D00/'
//SYSIN    DD *
<GIMUNZIP>
<ARCHDEF
name="S0005.USER.HMP1D00.DOCLIB.pax.Z"
newname="USERID.DOCUMENTS.HMP1D00" />
</GIMUNZIP>
/*
```

Figure 8-25 GIMUNZIP JCL

GIMZIP and GIMUNZIP availability

GIMZIP and GIMUNZIP services are available for current releases and can be used as follows:

- They enable use of GIMZIP packaging technology without requiring the latest release.
- GIMZIP packaging is being exploited by software available on the "Download Zone."
- Receive FromNet is not provided, only the GIMZIP and GIMUNZIP standalone services are available.
- Available on current releases via APAR IR43312
  - UR52471 OS/390 Version 2 Release 7, 8, 9, 10, and z/OS Release 1
  - UR52470 OS/390 Version 2 Release 5 and 6

9

Workload Manager enhancements

In this chapter, the following topics are discussed:

- LPAR CPU management for Linux
- Server task/thread management
- Crypto workload activity reporting
- Temporal affinity support
- Enclaves registration support
- Report class enhancements

9.1 LPAR CPU management for Linux

Intelligent Resource Director (IRD) functions shipped in z/OS V1R1 provide a method to move CPU resources to workloads across z/OS systems running in different logical partitions (LPARs). But an LPAR could not dynamically adjust CPU resource allocations for non-z/OS systems based on workload demands to meet goals. Therefore, a non-z/OS image could not take advantage of WLM goal-based dynamic CPU resource management. Non-z/OS systems include, for example, Linux, VM, OS/390, and VSE.

Now IRD supports non-z/OS images. IRD manages CPU resources for non-z/OS images based on WLM policy, and IRD moves CPU resources to workloads across z/OS and non-z/OS systems.

LPAR CPU management for Linux provides a flexibility in managing CPU resources across LPARs running Linux or other non-z/OS systems in accord with workload goals. WLM can shift weight from either z/OS or non-z/OS partitions to another z/OS or non-z/OS partition in the same LPAR cluster.

9.1.1 LPAR cluster

As part of LPAR CPU management support, which is a capability provided by z/OS V1R1, WLM has introduced the concept of an LPAR cluster. Currently, a central processing complex (CPC) can consist of several z/OS images that can be part of separate sysplexes. Each set of z/OS images within the same sysplex is designated as an entity. That entity is referred to as an LPAR cluster.

With LPAR CPU management for LINUX, a definition of LPAR cluster is broadened to include non-z/OS images in the cluster. Figure 9-1 shows a single sysplex defined to a CPC and three images. In this example, we have z/OS V1R2, Linux for zSeries, and VM/VIF images defined as part of the LPAR cluster.

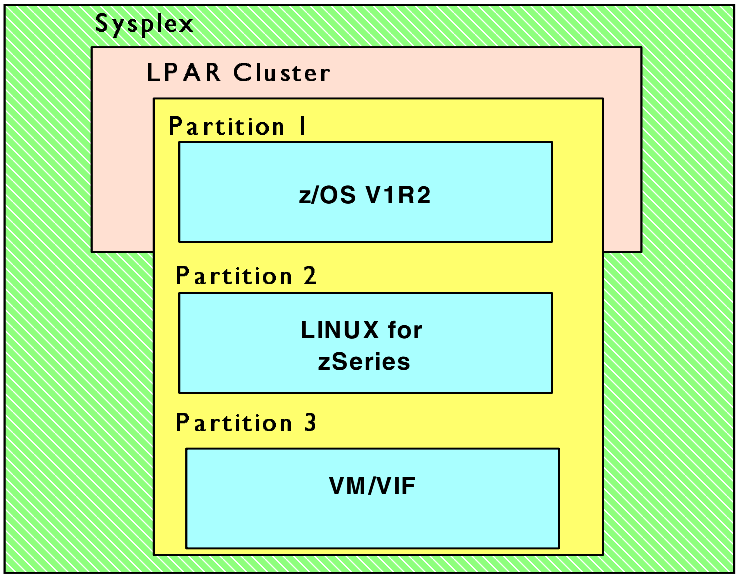


Figure 9-1 LPAR cluster in 1 CPC

Figure 9-2 on page 171 shows two sysplexes defined to a CPC. In sysplex 1, there are two images running z/OS V1R1 and V1R2 respectively, and they make up LPAR cluster 1. In sysplex 2, the images contain z/OS V1R2 and Linux for S/390 to make up LPAR cluster 2.

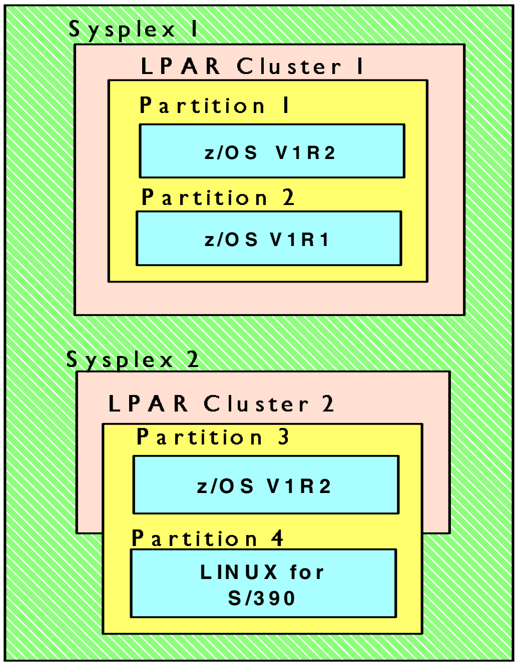


Figure 9-2 2 LPAR clusters in 1 CPC

Figure 9-3 is an example of multiple LPAR clusters in a SYSPLEX. LPAR cluster 1 includes the Linux partition in the cluster definition. The Linux partition on CPC 2 is defined to be excluded from LPAR cluster 2. Two OS/390 R10 systems are part of the sysplex, but neither is included in the LPAR cluster definition.

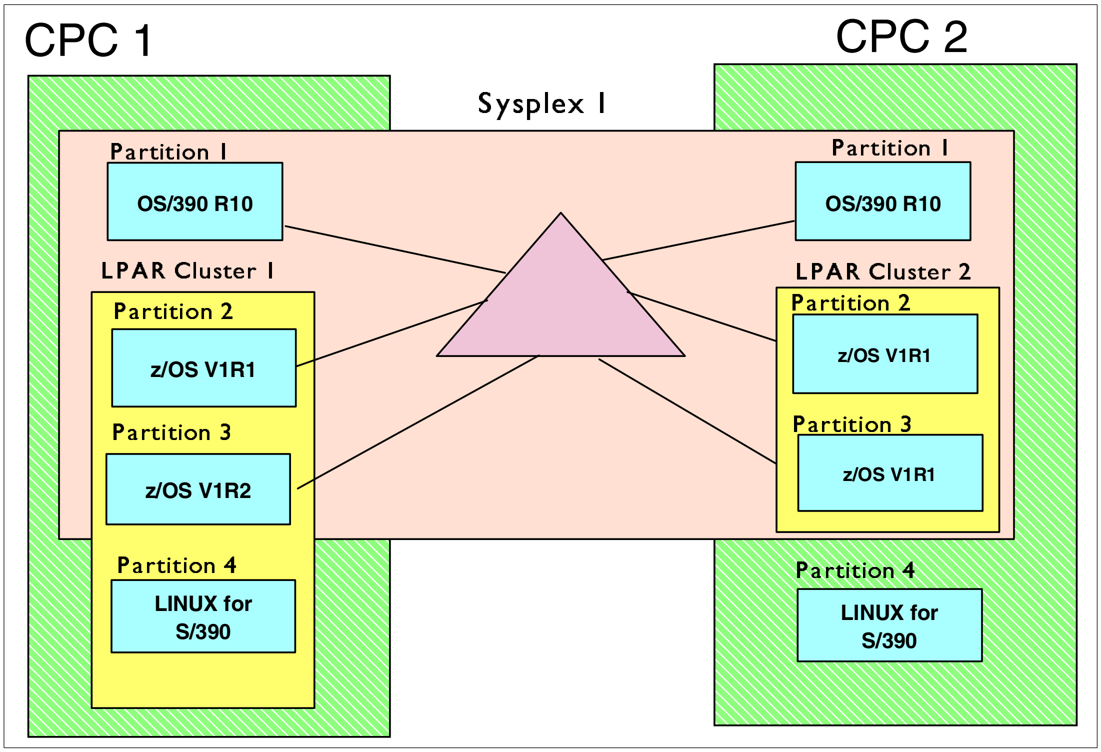


Figure 9-3 Multiple LPAR clusters in multiple CPCs

9.1.2 System requirements

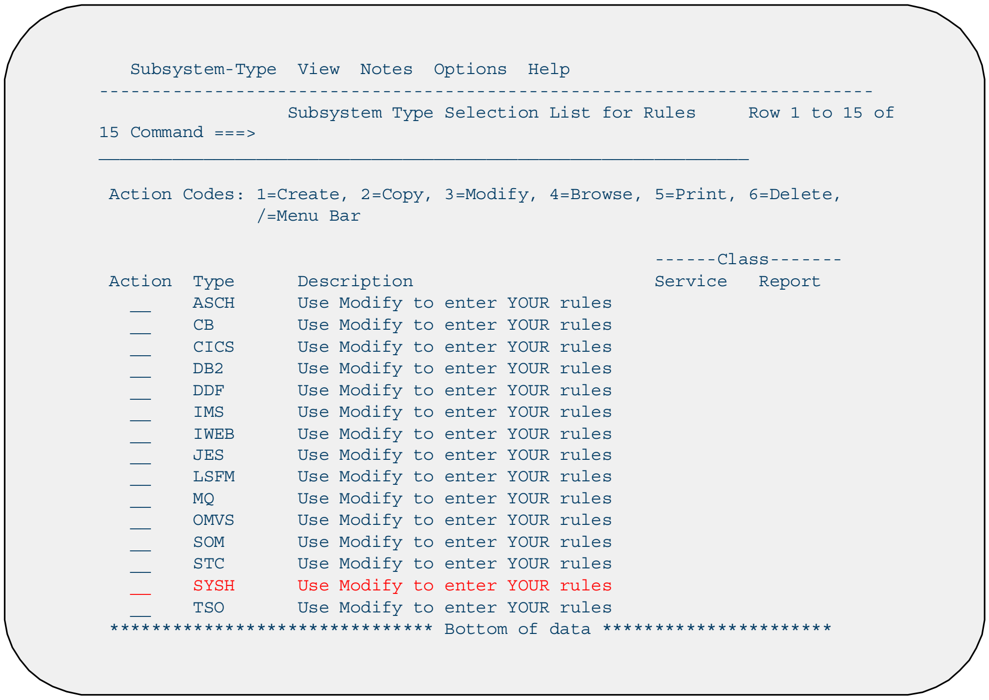
LPAR CPU management for Linux only works if:

- There is one z/OS V1R2 system with WLM in goal mode.
- The system is running Linux for S/390 or zSeries, kernel 2.4.
- CPC must be in a LPAR mode.
- Linux images must have shared CPs and must not be capped. Only standard general-purpose CPUs are supported; IFL (Integrated Facility for Linux) CPUs are not supported.
- Linux images must be “WLM Managed” in HMC logical partition control panel.
- IBM @server zSeries GA2 hardware is used.
- WLM CF structure is defined and connected.
- Goals are defined for Linux images in the WLM service definition.

9.1.3 Service definition changes

SYSH is a new subsystem type defined in a WLM administrative application to classify non-z/OS systems. See Figure 9-4 on page 173. Qualifier types allowed in SYSH are system names (SY and SYG) and sysplex name (PX). Service classes used in SYSH cannot be used in any other subsystem type. Also, it is recommended that each Linux partition have a separate service class unless Linux partitions are cloned and assumed to be running similar types of workloads.

Default service class is optional. It is recommended not to have a default service class, which will catch all unmatched non-z/OS systems. Report class is optional. To understand current workload characteristics running in a Linux partition, a report class can be used. A monitoring product like RMF reports the velocity of the workload running in a Linux partition, which can help customer to define the velocity goals for service classes associated with the Linux partition. More than one non-z/OS partition classified to the same service class is not recommended because WLM might not be able to manage the CPU resources across the non-z/OS partitions if the workload characteristics in these partitions varies.



```
   Subsystem-Type  View  Notes  Options  Help
--------------------------------------------------------------------------
                  Subsystem Type Selection List for Rules     Row 1 to 15 of
15 Command ===>
______________________________________________________________

 Action Codes: 1=Create, 2=Copy, 3=Modify, 4=Browse, 5=Print, 6=Delete,
               /=Menu Bar

                                                  ------Class-------
 Action  Type      Description                    Service   Report
   __    ASCH      Use Modify to enter YOUR rules
   __    CB        Use Modify to enter YOUR rules
   __    CICS      Use Modify to enter YOUR rules
   __    DB2       Use Modify to enter YOUR rules
   __    DDF       Use Modify to enter YOUR rules
   __    IMS       Use Modify to enter YOUR rules
   __    IWEB      Use Modify to enter YOUR rules
   __    JES       Use Modify to enter YOUR rules
   __    LSFM      Use Modify to enter YOUR rules
   __    MQ        Use Modify to enter YOUR rules
   __    OMVS      Use Modify to enter YOUR rules
   __    SOM       Use Modify to enter YOUR rules
   __    STC       Use Modify to enter YOUR rules
   __    SYSH      Use Modify to enter YOUR rules
   __    TSO       Use Modify to enter YOUR rules
 ******************************* Bottom of data **********************
```

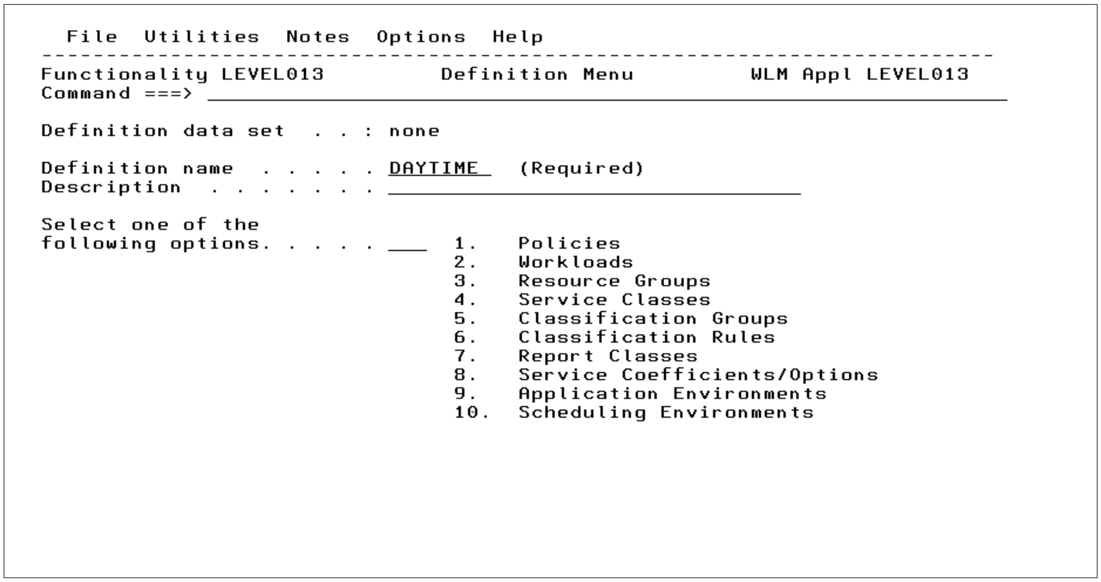
Figure 9-4 WLM subsystem types

Service classes used in SYSH have certain restrictions. Resource group is not allowed and the input field should be left blank. CPU critical is not supported and the input field should specify NO. Only single period velocity goal is allowed. Discretionary goal is not supported for the service classes.

An RMF workload activity report includes the service class and report class associated with a Linux partition. Velocity goals reported by RMF can be used to determine the velocity goal for a Linux partition.

9.1.4 Functionality level consideration

A service definition has a functionality level displayed on the administrative application definition menu panel. Level number increases if new functions are used. An administrative application can process any service definition at a functionality level equal to or less than the WLM application level. When a classification rule is using the SYSH subsystem type, the functionality level must be LEVEL013. Therefore, earlier systems cannot modify the service definition or cannot activate a service policy. Before it will become active, all restrictions and rules are introduced by functionality level LEVEL013. Figure 9-5 on page 174 shows the WLM administrative application definition menu panel. In addition, no migration or compatibility PTF is required.

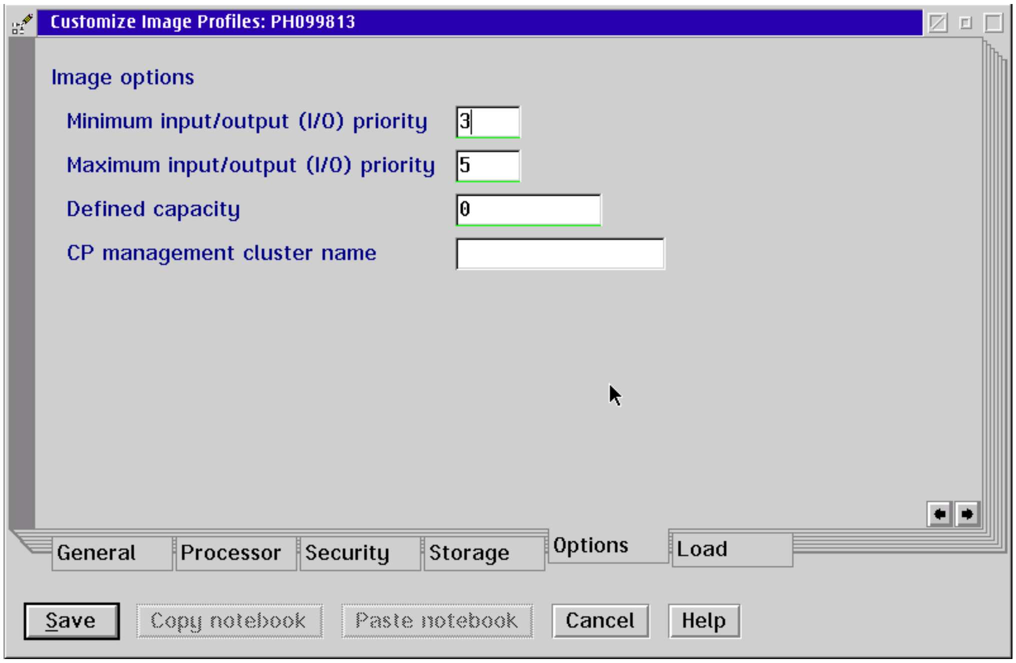


```
  File  Utilities  Notes  Options  Help
--------------------------------------------------------------------------
Functionality LEVEL013         Definition Menu          WLM Appl LEVEL013
Command ===> ______________________________________________________________

Definition data set  . . : none

Definition name  . . . . . DAYTIME   (Required)
Description  . . . . . . . ________________________________

Select one of the
following options. . . . . ___  1.   Policies
                                2.   Workloads
                                3.   Resource Groups
                                4.   Service Classes
                                5.   Classification Groups
                                6.   Classification Rules
                                7.   Report Classes
                                8.   Service Coefficients/Options
                                9.   Application Environments
                                10.  Scheduling Environments
```

Figure 9-5 WLM administrative application definition menu panel

9.1.5 Hardware HMC panel changes

A cluster name field is added to the partition definition in the hardware HMC panel, as shown in Figure 9-6. The cluster name for a Linux image should match with the sysplex name of z/OS V1.2 system. For z/OS images, it is not required to set the cluster name.



Customize Image Profiles: PH099813

Image options

Minimum input/output (I/O) priority: 3

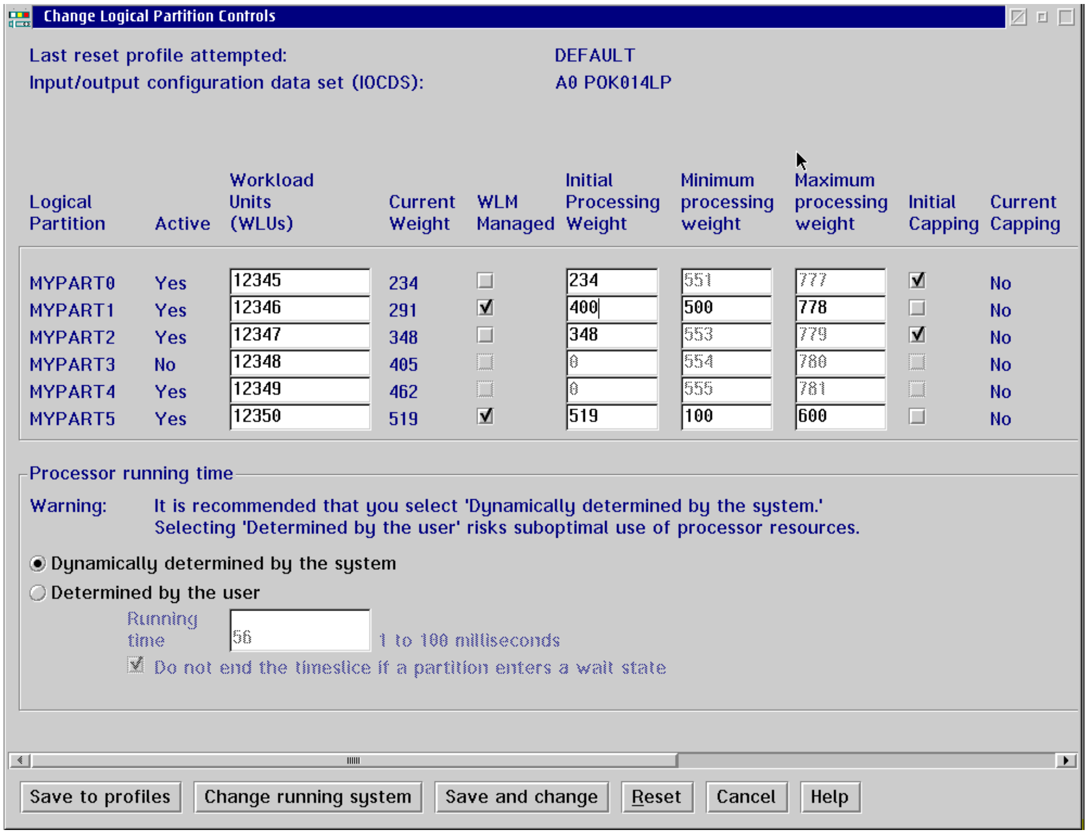
Maximum input/output (I/O) priority: 5

Defined capacity: 0

CP management cluster name:

Figure 9-6 Partition definition in hardware HMC panel

A partition weight value becomes an initial value. WLM keeps the sum of the weights of an LPAR cluster constant. This keeps the overall weight of the LPAR cluster constant compared to other partitions not in the cluster. WLM does not manage the LPAR weights of the Linux partitions that are capped, or have dedicated CPUs, or are running on IFL. If the current weight of a Linux partition is not the same as its initial weight, its weight could be adjusted when WLM is switched to compatibility mode on the other partition in the cluster. LPAR code resets the weight of the WLM-managed Linux partition to its initial weight if a partition leaves the cluster (system reset) or when WLM Management is disabled via the HMC panel. LPAR code also adjusts the weight of the Linux partition which is managed by WLM when the partition's minimum or maximum weight is changed. Figure 9-7 shows part of the hardware HMC logical partition controls panel.



Last reset profile attempted: DEFAULT
Input/output configuration data set (IOCDS): A0 POK014LP

Logical Partition	Active	Workload Units (WLUs)	Current Weight	WLM Managed	Initial Processing Weight	Minimum processing weight	Maximum processing weight	Initial Capping	Current Capping
MYPART0	Yes	12345	234	□	234	551	777	☑	No
MYPART1	Yes	12346	291	☑	400	500	778	□	No
MYPART2	Yes	12347	348	□	348	553	779	☑	No
MYPART3	No	12348	405	□	0	554	780	□	No
MYPART4	Yes	12349	462	□	0	555	781	□	No
MYPART5	Yes	12350	519	☑	519	100	600	□	No

Processor running time

Warning: It is recommended that you select 'Dynamically determined by the system.' Selecting 'Determined by the user' risks suboptimal use of processor resources.

● Dynamically determined by the system
○ Determined by the user

Running time 56 1 to 100 milliseconds
☑ Do not end the timeslice if a partition enters a wait state

Figure 9-7 Hardware HMC LPAR controls panel

9.1.6 How to work WLM and LPAR code

If goals cannot be met for a Linux partition:

1. Assess impact of increasing the partitions weight.
2. Find another z/OS or non z/OS partition in the cluster that can “donate weight” to the Linux partition.
3. Assess impact on work on the donor partition of reducing the partition's weight.
4. If the action makes sense based on the WLM policy, adjust the weights of the receiver and donor partitions.

9.1.7 LPAR CPU management for Linux example

Consider the following environment:

- 3 Workloads running on 1 CEC
  - OLTP is the most important workload. It runs only during the day shift.
  - Web server is running on Linux talking to DB2 on z/OS. It has medium importance.

  - Batch has low importance. It is always running and has a capability of consuming the whole CEC if allowed.
- The CEC is divided into 2 partitions:
  - Partition 1: z/OS running OLTP, DB2, and Batch
  - Partition 2: Web server on Linux
- IRD manage environment

Figure 9-8 shows the initial CPU resource distribution.

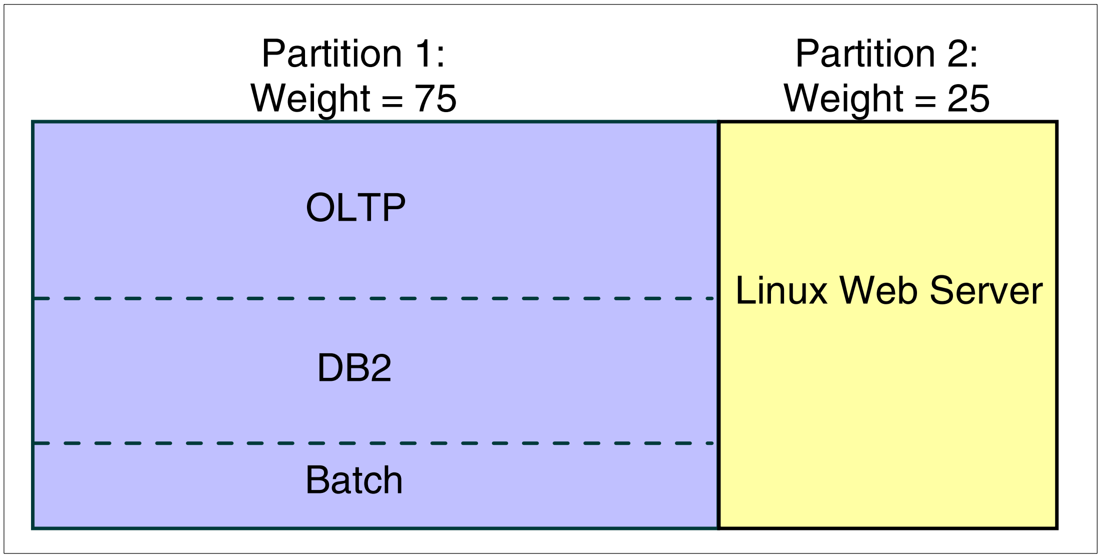


Figure 9-8 Initial CPU resource distribution

Then Figure 9-9 show a spike in OLTP work. Batch work is squeezed out and weights are adjusted to give more resource to partition 1.

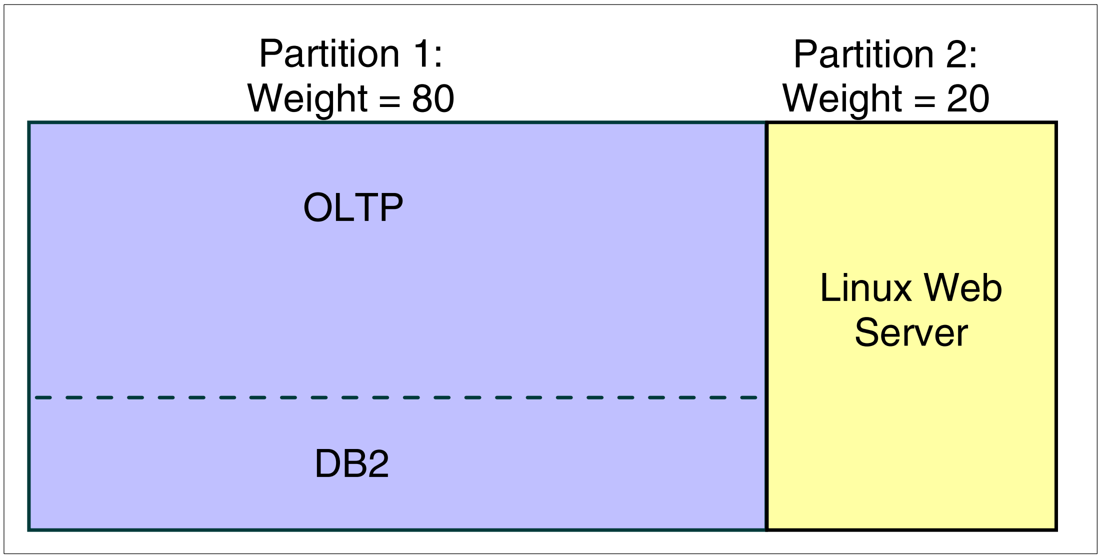


Figure 9-9 Spike in OLTP work

Finally, Figure 9-10 on page 177 shows the distribution when OLTP work ends. Weights are adjusted to give more resource to partition 2 and the resource given to batch is kept constant.

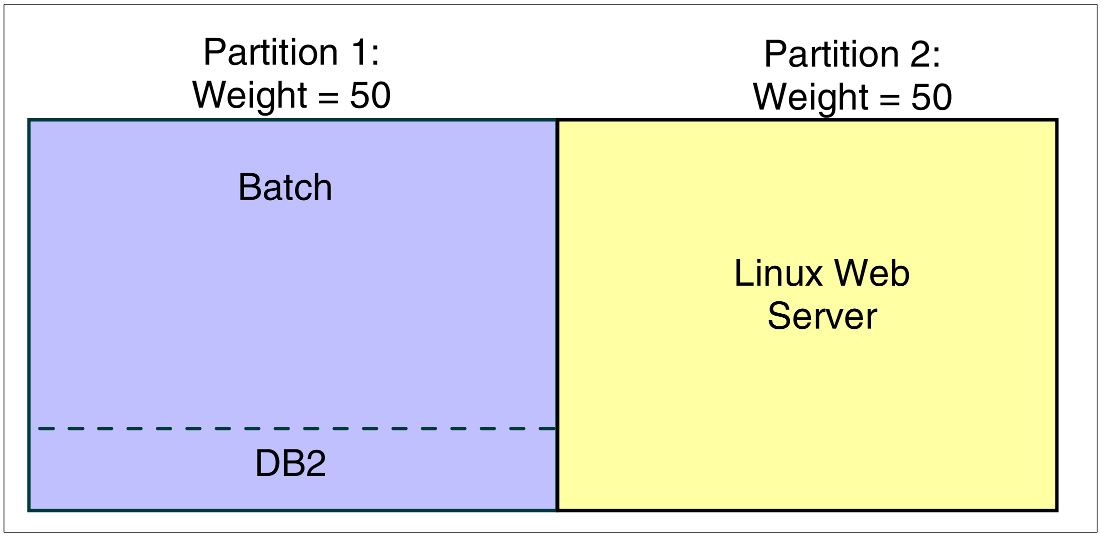


Figure 9-10 OLTP work ends

9.1.8 Enablement PTF

LPAR CPU management for Linux is shipped disabled in z/OS V1R2. To activate this function, an enablement PTF must be installed.

9.2 Server task/thread management

The queueing manager services are intended for queueing managers to use to manage server (execution) address spaces and the work requests which are processed to meet service class performance goals. Through queueing manager services, WLM maintains the queues for passing work requests from the queueing manager to its servers. A queueing manager is a subsystem that queues work requests to WLM for execution in server address spaces. Figure 9-11 on page 178 shows queueing manager services.

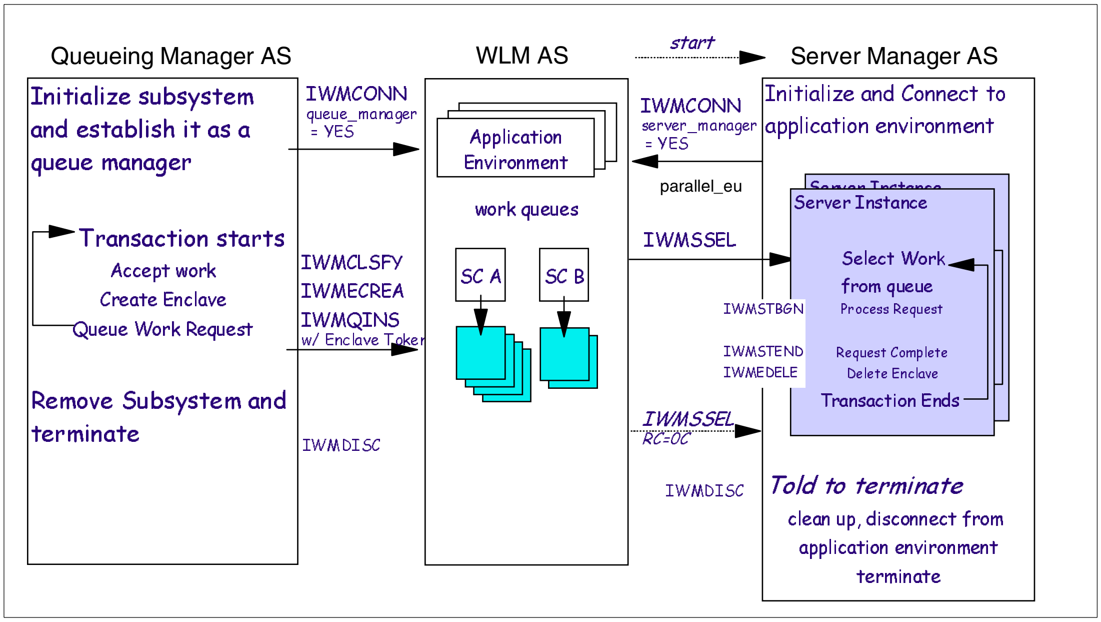


Figure 9-11 Queueing manager services

WLM dynamically starts and maintains server address spaces as required to meet the queueing manager's workload. Therefore, installations do not have to manage the server address spaces manually, nor do they have to monitor workload fluctuations that change the number of address spaces needed for the work to meet its goals. WLM automatically adjusts to changes in the workload.

A server address space contains one or multiple server instances which all select work requests from a work queue. The server address space tells WLM during startup how many server instances will be started (parameter parallel_eu on the IWMCONN service). All server address spaces of the same application environment must start with the same number of server instances.

Now WLM provides an interface that allows applications to obtain the number of server instances executing concurrently from WLM. WLM monitors and manages the number of server instances in a server address space.

9.2.1 Function

Server manager address space registers to let WLM manage the number of server instances. The IWMCONN interface changes to tell WLM that the server wants WLM to manage the server tasks. The server manager must listen for changes. IWMSINF is provided to obtain the number of server tasks to start. For server tasks, new return information on IWMSSEL is provided for stopping server tasks. See Figure 9-12 on page 179.

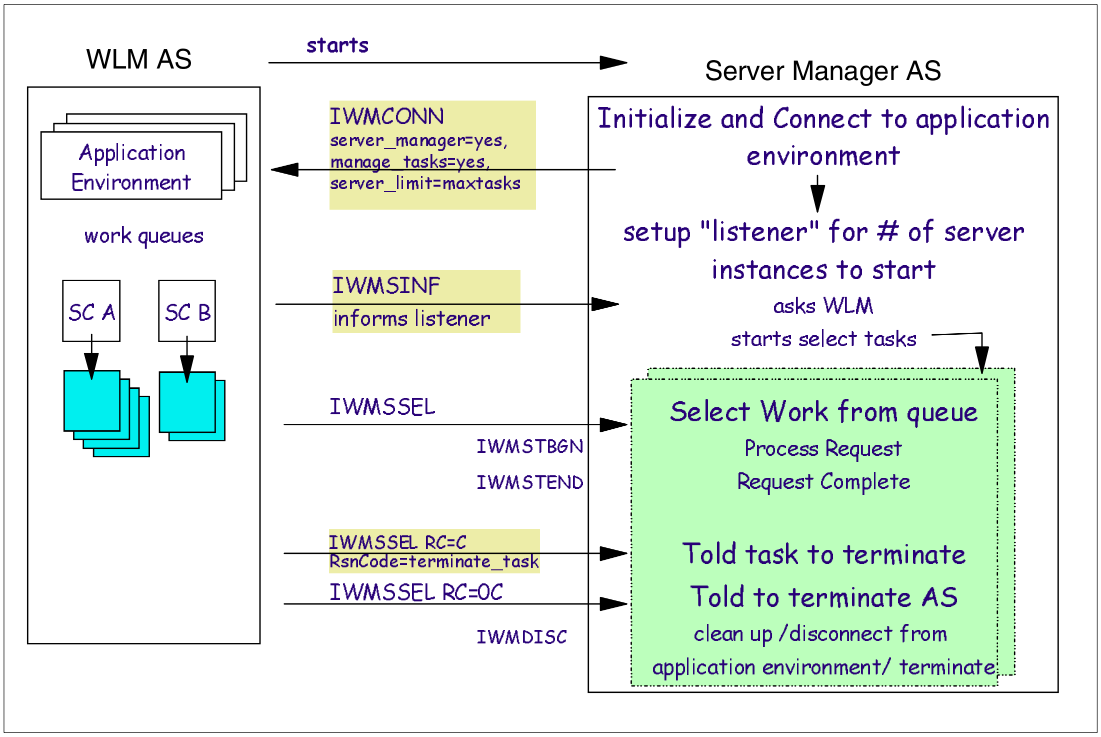


Figure 9-12 New Interfaces

The biggest advantage of the new function is that it eliminates the need for the end user to define the number of parallel execution units for the server address spaces. This task could be difficult and time-consuming for the customer. For the application programmer it was difficult, too, because the number of server instances might depend very much on end user characteristics which might not be known during development of the application. In addition, server task/thread management provides a better utilization of system resources because it reduces the need for starting too many server address spaces.

The criterion used to manage the number of server instances per server address space is the virtual storage consumption of the server tasks in the server address space. Therefore the function works best if a linear relationship between the number of server tasks and the virtual storage consumption exists.

9.3 Crypto workload activity reporting

Cryptography is an area of growing importance in an environment where more and more business is settled in electronic format. Cryptographic processing happens more or less invisibly to performance and capacity analysts. A solution is needed that provides more transparency for cryptographic processing. WLM samples address spaces or enclaves that are using any crypto-related hardware or that are waiting for such hardware.

9.3.1 CMOS cryptographic coprocessor

To understand the CMOS cryptographic coprocessor, it does not make any difference how it is implemented on a G5/G6 9672 enterprise server or on a zSeries 900 IBM @server.

The CMOS cryptographic coprocessor is an extension of a regular CPU. It consists of two parts: Direct Attached Crypto (DAC) and Cryptographic Asynchronous Message Processor (CAP). They have the following characteristics:

DAC DAC is designed for symmetric cryptographic algorithms, such as DES. It is executed synchronously; meaning that while the DAC is busy, the CPU is busy. No other instruction can be executed at the same time, until the DAC operation is completed. Furthermore, z/OS puts symmetric crypto requests intended for the DAC into a specific processor feature queue that is associated with the crypto CP, if the dispatcher selected a non-crypto CP in the first run. Work on such a feature queue is always dispatched prior to any other work.

CAP CAP is designed for asymmetric cryptographic algorithms, such as RSA. It is executed asynchronously to the CPU; meaning that while the CAP is busy, the CPU is ready for other work.

Therefore, there are two consequences for crypto workload reporting:

- Work that is executing synchronous cryptographic operations is either seen as CPU-using or CPU-delayed. Large amounts of synchronous crypto work are reflected by delays on a processor feature queue. WLM refers to such delays as feature queue delays.
- Work that is executing asynchronous cryptographic operations is transparent for the system and can only be estimated indirectly. WLM refers to samples related to the CAP as cryptographic asynchronous message (CAM) samples.

9.3.2 PCI cryptographic coprocessor (PCICC)

The Peripheral Component Interconnect Cryptographic Coprocessor (PCICC) is an orderable feature that adds additional cryptographic function and cryptographic performance to G5/G6 servers. Up to 8 PCICC features may be ordered for a G5 or G6 server. Support for PCICC was provided in OS/390 V2R9 by new ICSF functions. The PCICC feature will coexist with and augment the CMOS cryptographic coprocessor functions. ICSF will transparently route application requests for cryptographic services to one of the integrated cryptographic engines, either a CMOS cryptographic coprocessor or a PCICC, depending on performance or requested cryptographic function. For example, RSA signature generation and verification operations (with 1024-bit or shorter keys), such as those typically used by SSL, will be routed to both CMOS cryptographic coprocessor and PCICC engines. On the other hand, RSA key generation will be performed only on PCICC engines.

The z900 servers can also support up to 8 optional Dual Peripheral Component Interconnect Cryptographic Coprocessor (Dual PCICC) features. Each Dual PCICC feature contains 2 cryptographic coprocessors, for a maximum of 16 coprocessors. This provides the capability to support 2,000 SSL transactions/second. The Dual PCICC feature uses the same adapter card as used for the FICON and OSA-Express, and plugs into the I/O Cage.

Cryptographic requests directed to the PCICC or Dual PCICC are executed asynchronously to the CPU. WLM refers to samples related to PCICC as adjunct processor (AP) samples.

9.3.3 New general execution delay report samples

The IWMRCOLL service collects general execution delay data from the OUCB for address spaces or the ENCB for enclaves at certain events. It is called in case of

- Policy adjustment interval expiration
- Transaction start/stop/period switch
- On behalf of WLMCOLL sysevent (IRAWRCOL) to update in-flight transactions

The IWMRCOLL service return, IWMWRCAA, includes new general execution delay information which relates a service class period or a report class period as follows:

RcaeCryptoCAMU CAM crypto using samples. A task was found executing on a CAP.

RcaeCryptoCAMD CAM crypto delay samples. A task was found waiting for a CAP.

RcaeCryptoAPU AP crypto using samples. A task was found executing on a PCICC.

RcaeCryptoAPD AP crypto delay samples. A task was found waiting for a PCICC.

RcaeFeatureQD Feature queue delay samples. A task or SRB was found waiting on a processor feature queue associated with a CPU. This is a subset of CPU delay (RcaeCDEL).

Those new fields are set to zero in WLM compatibility mode.

9.3.4 New crypto using and delay report samples

IWMRQRY service return IWMWRQAA includes new general execution delay information, which relates an address space or an enclave:

RqaeCryptoCAMU CAM crypto using samples. A task was found executing on a CAP.

RqaeCryptoCAMD CAM crypto delay samples. A task was found waiting for a CAP.

RqaeCryptoAPU AP crypto using samples. A task was found executing on a PCICC.

RqaeCryptoAPD AP crypto delay samples. A task was found waiting for a PCICC.

RqaeFeatureQD Feature queue delay samples. A task or SRB was found waiting on a processor feature queue associated with a CPU. This is a subset of CPU delay (RqaeCDEL).

Those new fields are set to zero in WLM compatibility mode.

9.3.5 Crypto workload reporting

Crypto using and delay samples are collected on an address space or enclave basis. The samples are parts of the general execution delay samples, such as CPU using, idle, or unknown. However, they are reporting samples only and are not used for the WLM management algorithms.

The samples are aggregated by service class and report class associated with the address space or enclave, and are returned by WLM service IWMRCOLL. A current sampling snapshot for an address space or an enclave can be obtained by WLM service IWMRQRY that has been enhanced to also return a snapshot of crypto samples. Figure 9-13 on page 182 shows crypto-related WLM reporting services.

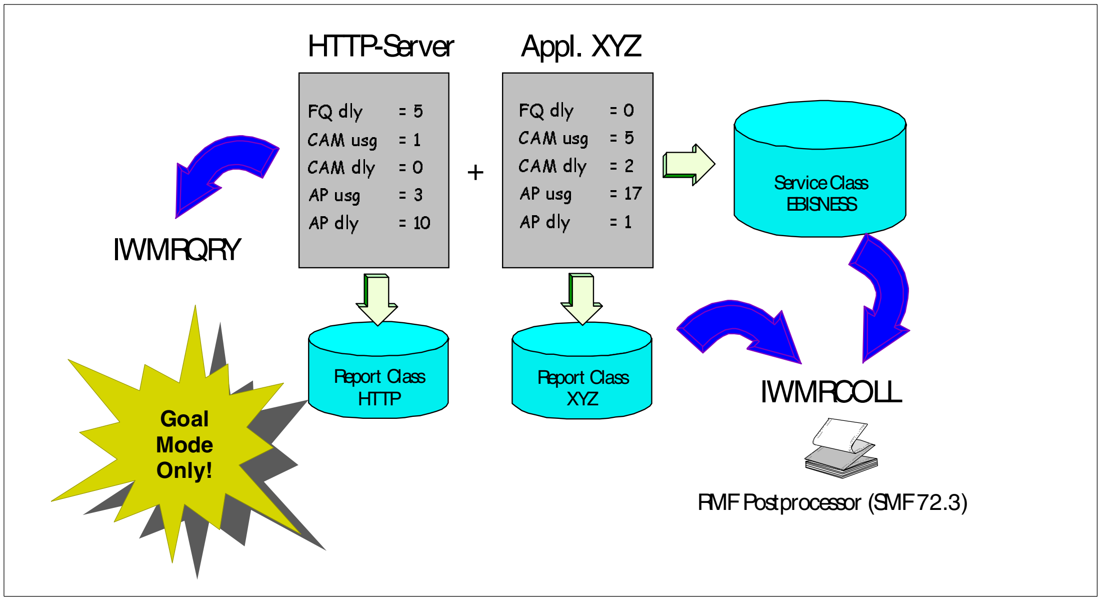


Figure 9-13 Crypto workload reporting services

CAM samples represent work that was found using the CAP or waiting for it. AP samples represent work that was found using the PCICC or waiting for it. Synchronous crypto delay represents work found on the processor feature queue. Work on the processor feature queue represents DAC only.

RMF stores the new crytpo samples in the SMF 72 subtype 3 record and reports them in the RMF WLMGL Postprocessor report:

- Aggregated crypto using samples percentage (CAM + AP).
- Aggregated crypto delay samples percentage (FQ + CAM + AP).
- Additionally, the individual using/delay percentage of FQ, CAM, or AP can be reported via RMF Overview Reporting.

See also “Crypto measurement reporting” on page 190.

9.4 Temporal affinity support

Existing workload manager interfaces allow a control region to queue work requests to a pool of server regions for a service class. The assumption is that a transaction starts with inserting the work requests and ends with leaving (potentially deleting) the enclave, although there are cases where information must be preserved across multiple “independent” work requests and the information left behind lives only in the virtual storage of the address space. Following work requests requiring this information must be directed to the server region which has this information.

With WLM temporal affinity support, a server region is able to obtain a region token at a select time in which IWMSSEL is issued. The server region passes this region token to a queue manager. The queue manager is now able to route subsequent requests directly to the server region by specifying the region token on IWMQINS. In addition to this, new IWMTAFF service allows the server or control region to mark the server region as being needed by follow-on work requests. It is an extension of the insert mechanism to allow the control region to direct a work request to a specific server region. WLM will ensure that server region stays alive until all temporal affinities have been removed.

Figure 9-14 shows a temporal affinity support overview. For complete information, the figure also shows the ability of the queue manager to queue directly to a server task represented by a server token (dotted line and dotted queue symbol). This feature is used if it is necessary to exchange information between the client and the application within the scope of a transaction. It was introduced with OS/390 R5 and is also exploited by WebSphere. In contrast, temporal affinity support, which queues to the region, allows you to exchange or keep information across multiple transactions.

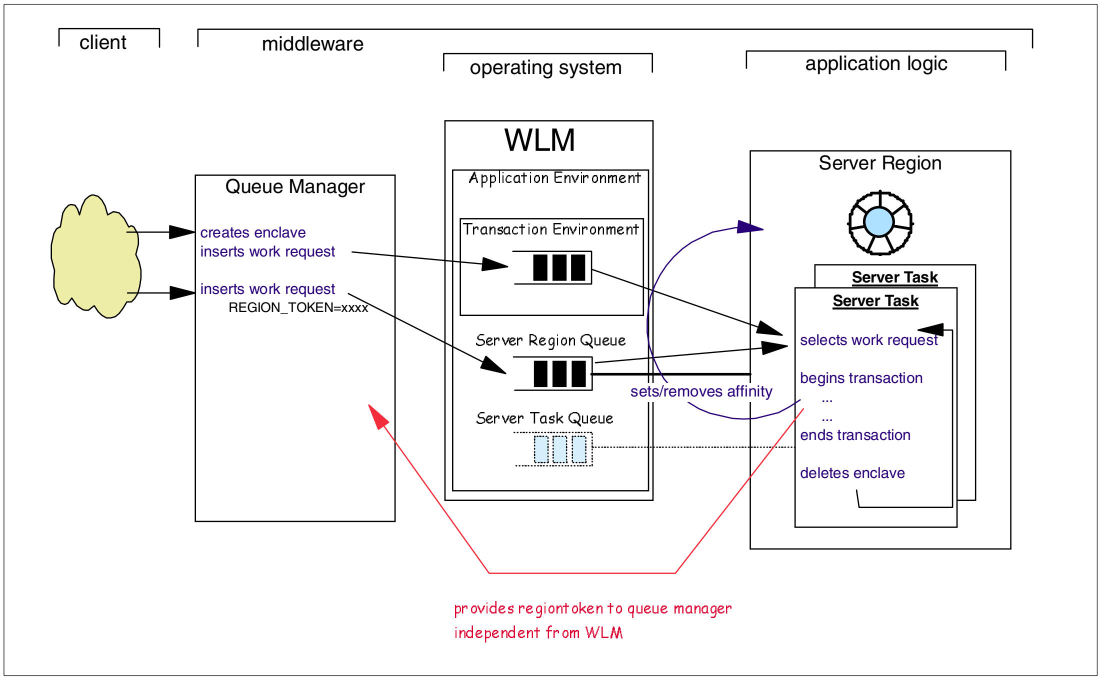


Figure 9-14 Temporal affinity support overview

9.4.1 System requirements

Hardware	None.
Software	OS/390 V2R8 or higher with APAR OW45238. Integrated in z/OS V1R2.
Exploiter	WebSphere V4. For further dependencies see WebSphere V4 documentation.

9.4.2 MVS message/SRM/RMF/SDSF

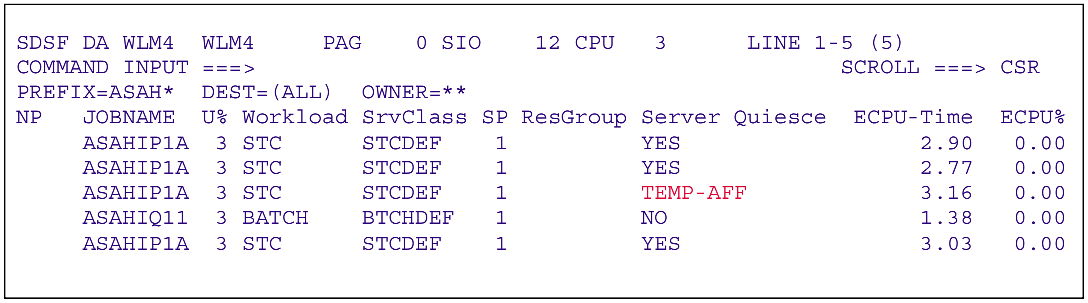
If temporal affinity is used by applications like WebSphere, it is no longer possible to terminate a server region when such a temporal affinity exists. The server region uses a new interface IWMTAFF to tell WLM that it has an affinity to a client. It is also the responsibility of the server region to tell WLM that the affinity no longer exists.

As long as the server region has an affinity, WLM will not terminate the server address space. This is related to the WLM refresh and quiesce vary commands, which terminate the existing server regions. The command is being processed, but server regions which indicate affinities to clients are not stopped until the affinities are discontinued.

In order to inform the operator, a message is issued that the refresh or quiesce is in progress and that a temporal affinity exists. The message also displays the systems on which server regions with affinities run.

```
IWM031I  15.33.28 VARY [REFRESH|QUIESCE] FOR appl. env. IN PROGRESS[. TEMPORAL
AFFINITIES EXIST ON wlm1[,wlm2]]
```

For further investigations and for determining which server regions have affinities, the SDSF display active screen has been enhanced; see Figure 9-15. The SDSF display active screen has been enhanced to tell the operator not only that an address space is managed as server (YES in column Server) but also whether a temporal affinity exists (TEMP-AFF). The value temp-aff includes that the server region is managed as a server.



```
SDSF DA WLM4  WLM4     PAG    0 SIO    12 CPU    3      LINE 1-5 (5)
COMMAND INPUT ===>                                            SCROLL ===> CSR
PREFIX=ASAH*  DEST=(ALL)  OWNER=**
NP   JOBNAME  U% Workload SrvClass SP ResGroup Server Quiesce  ECPU-Time  ECPU%
     ASAHIP1A  3 STC      STCDEF    1          YES                  2.90   0.00
     ASAHIP1A  3 STC      STCDEF    1          YES                  2.77   0.00
     ASAHIP1A  3 STC      STCDEF    1          TEMP-AFF             3.16   0.00
     ASAHIQ11  3 BATCH    BTCHDEF   1          NO                   1.38   0.00
     ASAHIP1A  3 STC      STCDEF    1          YES                  3.03   0.00
```

Figure 9-15 SDSF display active screen example

Other products (for example monitors like RMF monitor II) can use the REQASD and REQFASD sysevents to obtain information about address spaces. SRM returns an indication whether an address space has a temporal affinity. RMF monitor II saves this information in the SMF type 79 record subtypes 1 and 2 and SDSF obtains it through the ERB2XDGS interface from RMF.

While WLM does not automatically stop an address space which indicates a temporal affinity, the operator can still cancel such an address space.

9.5 Enclave registration support

Enclaves were introduced in MVS 5.2.0 to represent business transactions or units of work. Enclave transactions map closely to the end user's view of a business transaction and usually comprise multiple executable pieces (tasks or SRBs) that run in one or more address spaces. Every task or SRB in that enclave is managed by the system towards the goal defined for this specific type of business transaction.

Anyway, enclave transactions do not only exist within a single subsystem but also across subsystems. The subsystems can decide if and how they communicate enclave tokens as transactions flow across subsystem boundaries. This structure requires a tight synchronization by all involved subsystems. Today, enclaves can be deleted by any subsystem at any time. It is the responsibility of the subsystems to prevent this while work is still being executed for the enclave. However, this is not always possible, depending on the architecture of the application.

Consider, for example, the following scenario:

An HTTP server receives a database query from the network. It creates an enclave and calls DB2 while running under this enclave. DB2 starts to process the query by scheduling SRBs into the enclave or joining tasks. If at some point the HTTP server terminates and deletes the enclave, any attempt of DB2 to further use the enclave will fail.

WLM has been enhanced to avoid premature deletion of an enclave. A “registration” concept has been introduced to allow subsystems to express interest in an enclave. The registration of interest in an enclave allows WLM to recognize the situation that one or more subsystems are still using an enclave.

From the point of view of the enclave's owner, the enclave delete service will continue to work as in the past. The service returns immediately, but physical deletion of the enclave (and the invalidation of the enclave token) is deferred by WLM until all registrations have been withdrawn. An enclave which is deleted while it is still registered is “logically deleted” only.

For multisystem enclaves, registration may also be used. It protects the local copy of the multisystem enclave only, as by registering the original or one of the foreign enclaves a subsystem ensures that this local enclave will not be deleted.

DB2 V7 exploits the new registration services to allow rollback and recovery processing to use the enclave created for the original transaction. Thus they are able to maintain the correct performance context for this transaction.

Any other IBM and non-IBM subsystem that shares enclaves with other subsystems might consider using the new services to avoid tight synchronization protocols and to simplify their processing. It also may be used to maintain the same performance context for all stages of a single transaction.

9.5.1 Registration services

Two new services are introduced:

IWMEREG	Register enclave
IWMEDREG	Deregister enclave

The new service IWMEREG allows a task or SRB to register a specific enclave it intends to use. IWMEREG takes the enclave token as input, validates the user input and returns a registration token upon successful registration of the enclave. The registration token must be saved by the caller to properly deregister the enclave at a later point in time. While an enclave is registered, the enclave cannot be physically deleted. If an attempt is made to delete the enclave anyway, it is set to a “logically deleted” state which does not restrict the functionality

The new service IWMEDREG allows a task or SRB to deregister a specific enclave and to enable physical deletion of the enclave. The caller passes the enclave token and the registration token that was obtained previously from IWMEREG to IWMEDREG to identify the enclave it wants to deregister. If there are no more registrations left for an enclave, a call of IWMEDELE would cause immediate deletion of the enclave. If IWMEDREG is invoked to deregister an enclave that is already logically deleted, the system completes the deletion automatically.

9.6 Report class enhancements

Report classes provide the means to report on any subset of the system's workload regardless of one or more service classes. Typical reporting goals for report classes are:

- To summarize by department

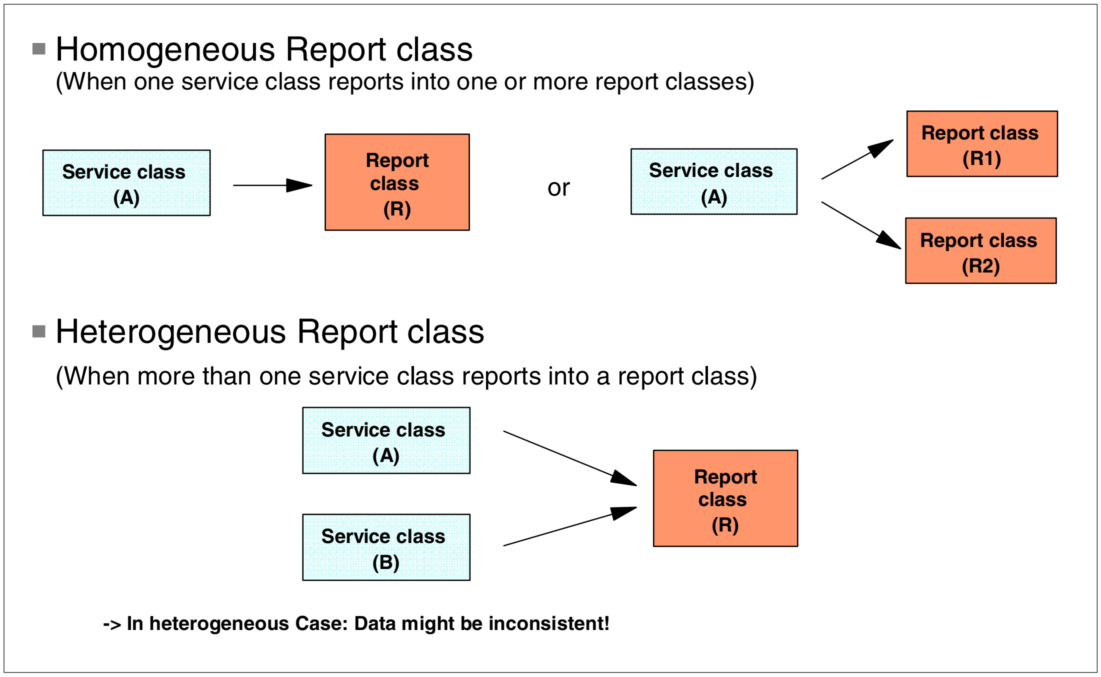
- To summarize by organization
- To break down reporting for a single TSO user

Today report classes do not provide the notion of periods, and report classes for work associated with response time goals do not provide response time distribution information (only service classes do so). Also, report classes for work associated with monitoring environments do not provide work manager delay states (sampling data only in service classes).

With WLM report class enhancements, report classes are extended to provide the missing capabilities described in the preceding example. In addition, reporting interfaces are extended to tell the reporting tool whether the information provided for report class is homogeneous or heterogeneous.

9.6.1 Homogeneous/heterogeneous

Report classes are defined with the WLM administration application and are associated with single transactions or groups of transactions by applying one or more classification rules. Two types of relationships between one or more service classes and one or more report classes exist. One is a homogeneous relationship when one service class reports into one or more report classes. The other is a heterogeneous relationship when more than one service class report into a report class. Figure 9-16 shows a homogeneous report class and a heterogeneous report class.



Homogeneous Report class
(When one service class reports into one or more report classes)
Service class (A)
Report class (R)
or
Service class (A)
Report class (R1)
Report class (R2)
Heterogeneous Report class
(When more than one service class reports into a report class)
Service class (A)
Service class (B)
Report class (R)
-> In heterogeneous Case: Data might be inconsistent!

Figure 9-16 Report classes: homogeneous and heterogeneous

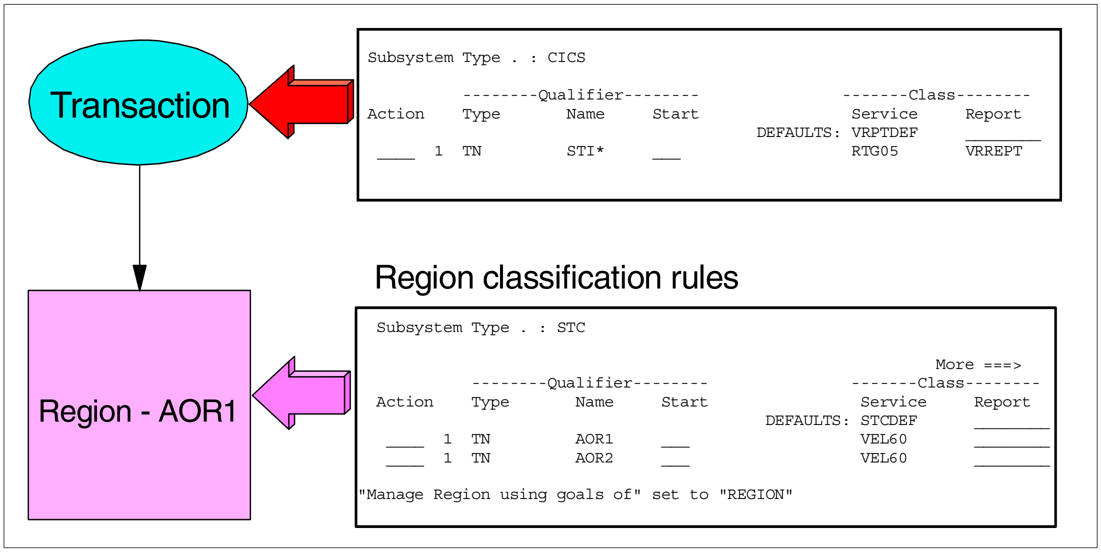
We should note that the data might be inconsistent in heterogeneous report classes, as goals of related service classes might differ in type, periods, or values.

9.6.2 Usage example

This example is a goal mode migration for granularity of CICS/IMS region management.

With OS/390 V2R10, a customer can tell WLM to manage a CICS/IMS transaction either to its transaction goal or to the goals of its serving regions. If CICS/IMS transactions are managed to region goals, no workload reporting data will be contributed into the service class of the CICS/IMS transaction. But the reporting data will be contributed into its report class, if assigned in the classification rules. Figure 9-17 shows a classification example.

.



```
Subsystem Type . : CICS

          --------Qualifier--------                    -------Class--------
Action    Type        Name      Start                   Service      Report
                                            DEFAULTS: VRPTDEF      ________
 ____  1  TN          STI*      ___                     RTG05        VRREPT
```

```
Subsystem Type . : STC

                                                                 More ===>
          --------Qualifier--------                    -------Class--------
Action    Type        Name      Start                   Service      Report
                                            DEFAULTS: STCDEF       ________
 ____  1  TN          AOR1      ___                     VEL60        ________
 ____  1  TN          AOR2      ___                     VEL60        ________

"Manage Region using goals of" set to "REGION"
```

Figure 9-17 Granularity of CICS/IMS region management

With the full reporting scope provided by the report class enhancements, customers can now gather response time distribution data for CICS/IMS transactions while the region is still controlled by velocity goals.

9.6.3 Interaction with RMF

See “WLM report class periods support” on page 193.

10

RMF enhancements

This chapter describes the enhancements to RMF in support of the changes in z/OS Version 1 Release 2 in the following areas:

- Crypto measurement reporting
- Coupling facility duplexing support
- WLM report class periods
- Defined capacity support for sub-capacity Workload License Charges (WLC) pricing
- LDAP integration
- 3270 local session removal

10.1 Crypto measurement reporting

Up to 8 PCI cryptographic coprocessors (PCICC) can be plugged into available slots in the IBM @server zSeries to accelerate encrypting and decrypting. The RMF Crypto Hardware Activity Report providing data about the installed crypto hardware can be generated as an RMF monitor I real-time report or as a postprocessor report by the new report option REPORTS(CRYPTO). The report will only be available for IBM @server zSeries GA2.

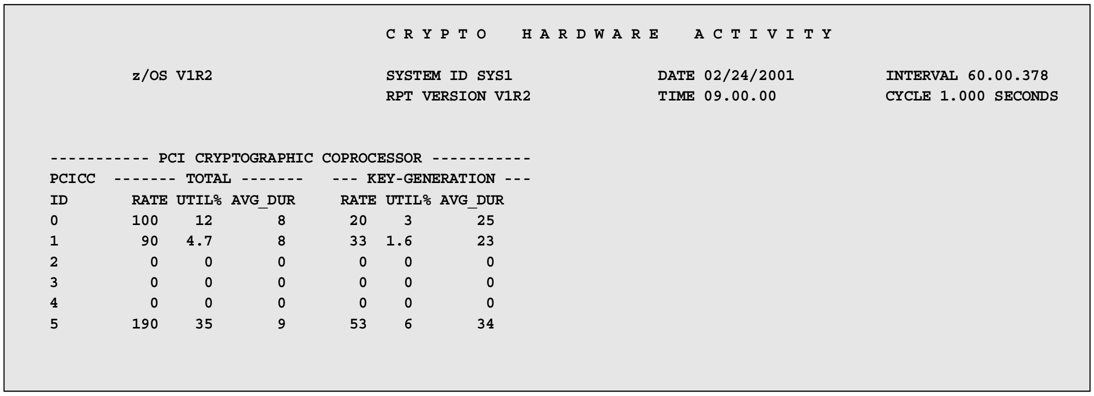
See Figure 10-1. In order to evaluate the utilization of the PCICCs, 3 kinds of fields are provided:

Request rate Requests/second

Utilization % Busy time/interval time * 100

Average Duration The amount of time (in ms) for a single request

In addition to standard encrypting/decrypting requests, key generation requests are registered separately and an additional section shows the counts for this specific subset of requests.



```
                          C R Y P T O    H A R D W A R E    A C T I V I T Y

      z/OS V1R2                 SYSTEM ID SYS1            DATE 02/24/2001        INTERVAL 60.00.378
                                RPT VERSION V1R2          TIME 09.00.00          CYCLE 1.000 SECONDS


---------- PCI CRYPTOGRAPHIC COPROCESSOR ----------
PCICC  ------- TOTAL -------   --- KEY-GENERATION ---
ID       RATE UTIL% AVG_DUR     RATE UTIL% AVG_DUR
0         100    12       8       20    3       25
1          90   4.7       8       33  1.6       23
2           0     0       0        0    0        0
3           0     0       0        0    0        0
4           0     0       0        0    0        0
5         190    35       9       53    6       34
```

Figure 10-1 Crypto Hardware Activity Report sample

The data which has to be gathered for the crypto hardware is kept in the new SMF record type 70 subtype 2. The gathering can be switched on/off by the option CRYPTO/NOCRYPTO

For capacity planning purposes, the following new overview control statements are introduced:

PCSNRRQ Request rate

PCSBUSY Utilization %

Workloads can be impacted by the limited availability of cryptographic processor power. Therefore, WLM will introduce the sampling of address spaces or enclaves that are using or waiting for crypto-related hardware. For further discussion, see “Crypto workload activity reporting” on page 179. These report samples will be formatted by the RMF Workload Activity Goal Mode Report. Figure 10-2 on page 191 shows an RMF Workload Activity Report sample.

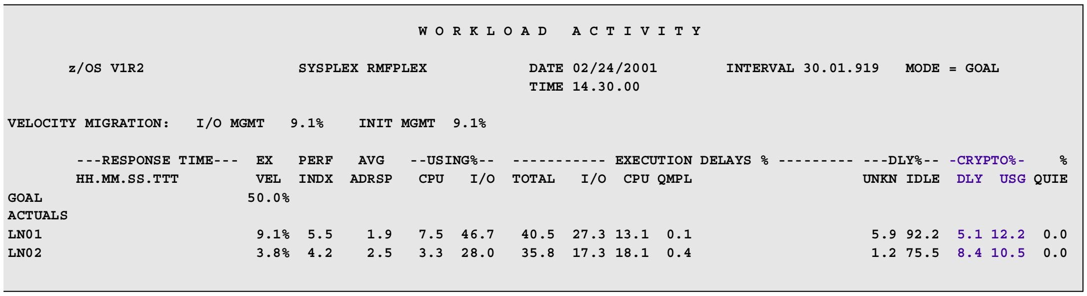
There are 3 different types of delay/using samples:

CAM crypto samples A TCB was found executing on or waiting for a Cryptographic Asynchronous Message Processor (CAP).

Feature Queue samples A TCB was found waiting for a processor feature queue associated with a CPU.

AP Crypto samples A TCB was found executing on or waiting for a cryptographic Adjunct Processor.

While CAM and FQs are related to CMOS cryptographic coprocessors, APs are related to the new PCI crypto coprocessor hardware. In contrast to the fields of the RMF Crypto Hardware Activity Report, all kinds of using and delay counts are collected with z/OS V1R2 on the existing enterprise servers G5, G6 and IBM @serverzSeries.



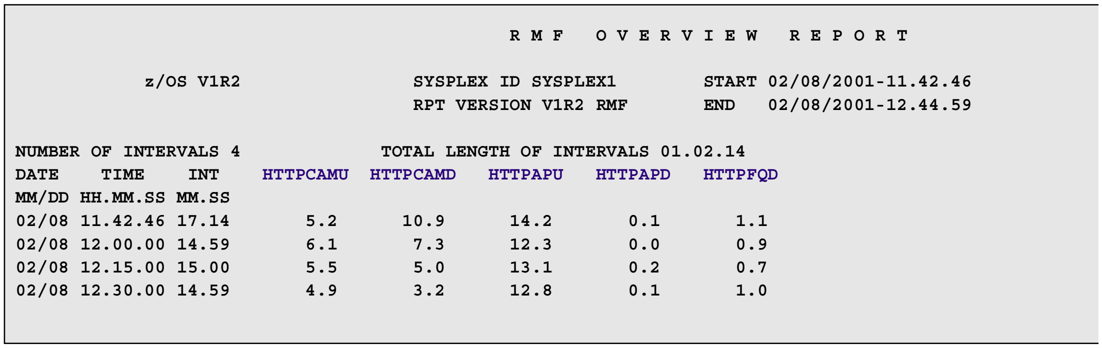
```
                                            W O R K L O A D   A C T I V I T Y

       z/OS V1R2                  SYSPLEX RMFPLEX            DATE 02/24/2001        INTERVAL 30.01.919   MODE = GOAL
                                                             TIME 14.30.00

VELOCITY MIGRATION:   I/O MGMT   9.1%    INIT MGMT  9.1%

        ---RESPONSE TIME---  EX    PERF   AVG   --USING%--  ----------- EXECUTION DELAYS % --------- ---DLY%-- -CRYPTO%-    %
        HH.MM.SS.TTT        VEL   INDX  ADRSP   CPU   I/O   TOTAL   I/O  CPU QMPL                    UNKN IDLE  DLY  USG QUIE
GOAL                       50.0%
ACTUALS
LN01                        9.1%  5.5    1.9    7.5  46.7   40.5  27.3 13.1  0.1                      5.9 92.2  5.1 12.2  0.0
LN02                        3.8%  4.2    2.5    3.3  28.0   35.8  17.3 18.1  0.4                      1.2 75.5  8.4 10.5  0.0
```

Figure 10-2 Workload Activity Report sample

Since the WLMGL report does not show the using/delays on type granularity, the following overview criteria are introduced for more detailed reporting:

CAPUSGP	Crypto CAP using%
CAPDLYP	Crypto CAP delay%
FQDLY	Crypto FQ delay%
APUSGP	Crypto AP using%
APDLYP	Crypto AP delay%
CRYUSGP	Crypto overall using%
CRYDLYP	Crypto overall delay%

Figure 10-3 shows an RMF Overview Report sample related to crypto control statements.

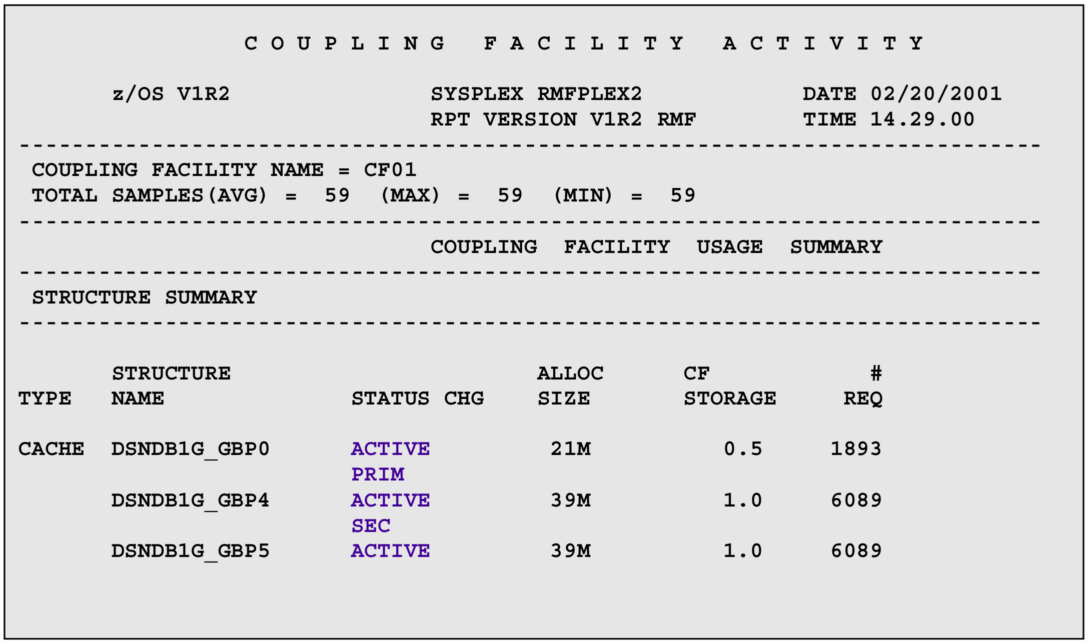


```
                                         R M F   O V E R V I E W   R E P O R T

            z/OS V1R2                SYSPLEX ID SYSPLEX1        START 02/08/2001-11.42.46
                                     RPT VERSION V1R2 RMF       END   02/08/2001-12.44.59

NUMBER OF INTERVALS 4              TOTAL LENGTH OF INTERVALS 01.02.14
DATE    TIME     INT   HTTPCAMU  HTTPCAMD   HTTPAPU   HTTPAPD   HTTPFQD
MM/DD HH.MM.SS MM.SS
02/08 11.42.46 17.14       5.2      10.9      14.2       0.1       1.1
02/08 12.00.00 14.59       6.1       7.3      12.3       0.0       0.9
02/08 12.15.00 15.00       5.5       5.0      13.1       0.2       0.7
02/08 12.30.00 14.59       4.9       3.2      12.8       0.1       1.0
```

Figure 10-3 RMF overview report sample

10.2 Coupling facility duplexing support

z/OS V1R2 provides support for system-managed duplexing. Creating a duplex copy of the structure in advance of any failure, and then maintaining the structure in a synchronized duplexed state during normal mainline operation, provides a robust failure recovery capability through failover to the unaffected structure instance. The recent RMF CF reporting enhancements provide the possibility to monitor and evaluate the actual CF structure placement and performance for duplexed structures, and make any necessary tuning changes. See Figure 10-4.



```
                 C O U P L I N G   F A C I L I T Y   A C T I V I T Y

       z/OS V1R2               SYSPLEX RMFPLEX2            DATE 02/20/2001
                               RPT VERSION V1R2 RMF        TIME 14.29.00
-----------------------------------------------------------------------------
 COUPLING FACILITY NAME = CF01
 TOTAL SAMPLES(AVG) =  59  (MAX) =  59  (MIN) =  59
-----------------------------------------------------------------------------
                               COUPLING  FACILITY  USAGE  SUMMARY
-----------------------------------------------------------------------------
 STRUCTURE SUMMARY
-----------------------------------------------------------------------------

       STRUCTURE                       ALLOC      CF            #
TYPE   NAME              STATUS CHG    SIZE       STORAGE     REQ

CACHE  DSNDB1G_GBP0      ACTIVE          21M          0.5     1893
                         PRIM
       DSNDB1G_GBP4      ACTIVE          39M          1.0     6089
                         SEC
       DSNDB1G_GBP5      ACTIVE          39M          1.0     6089
```

Figure 10-4 CF activity report summary section sample

The STATUS column of the Structure Summary section has been extended to indicate the duplexing attributes for a specific structure:

ACTIVE PRIM The structure is the old (primary) structure in a duplexing.

ACTIVE SEC The structure is the new (secondary) structure in a duplexing.

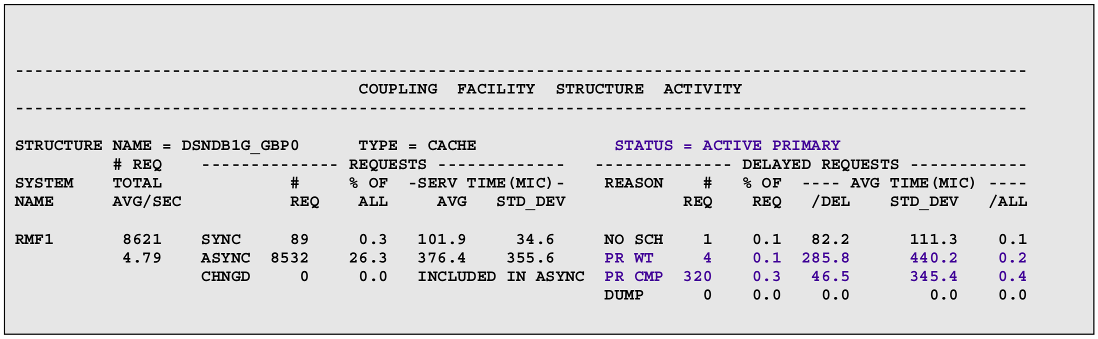
10.2.1 CF structure activity reporting

New duplexing-related delay reasons are shown in the structure activity section:

PR WT A subchannel for the operation targeted to the peer structure was not available.

PR CMP One of the two duplexed operations has completed, but the completed subchannel remains unavailable for use until the peer operation completes.

An example is shown in Figure 10-5 on page 193.

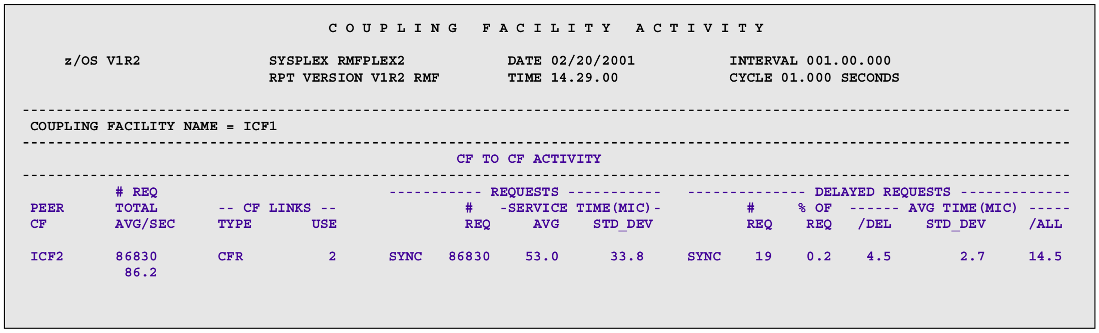


```
---------------------------------------------------------------------------------------------------------
                                   COUPLING  FACILITY  STRUCTURE  ACTIVITY
---------------------------------------------------------------------------------------------------------

STRUCTURE NAME = DSNDB1G_GBP0      TYPE = CACHE               STATUS = ACTIVE PRIMARY
          # REQ    -------------- REQUESTS -------------    -------------- DELAYED REQUESTS -------------
SYSTEM    TOTAL              #     % OF  -SERV TIME(MIC)-    REASON    #   % OF  ---- AVG TIME(MIC)  ----
NAME      AVG/SEC           REQ     ALL     AVG   STD_DEV            REQ    REQ   /DEL    STD_DEV    /ALL

RMF1       8621    SYNC      89     0.3   101.9      34.6    NO SCH    1    0.1   82.2      111.3     0.1
           4.79    ASYNC   8532    26.3   376.4     355.6    PR WT     4    0.1  285.8      440.2     0.2
                   CHNGD      0     0.0   INCLUDED IN ASYNC  PR CMP  320    0.3   46.5      345.4     0.4
                                                             DUMP      0    0.0    0.0        0.0     0.0
```

Figure 10-5 CF activity report subchannel activity section sample

10.2.2 CF-to-CF activity reporting

The new CF-to-CF activity section shows a summary of basic counts for system managed duplexing related operations only. The structure of the section is derived from the existing subchannel activity section and shows basically the same fields. See Figure 10-6.



```
                                    C O U P L I N G   F A C I L I T Y   A C T I V I T Y

     z/OS V1R2               SYSPLEX RMFPLEX2           DATE 02/20/2001           INTERVAL 001.00.000
                             RPT VERSION V1R2 RMF       TIME 14.29.00             CYCLE 01.000 SECONDS

--------------------------------------------------------------------------------------------------------------------------
 COUPLING FACILITY NAME = ICF1
--------------------------------------------------------------------------------------------------------------------------
                                                  CF TO CF ACTIVITY
--------------------------------------------------------------------------------------------------------------------------
           # REQ                           ----------- REQUESTS -----------  -------------- DELAYED REQUESTS -------------
 PEER      TOTAL       -- CF LINKS --              #    -SERVICE TIME(MIC)-          #     % OF  ------ AVG TIME(MIC) -----
 CF        AVG/SEC     TYPE       USE              REQ      AVG   STD_DEV            REQ    REQ   /DEL    STD_DEV     /ALL

 ICF2      86830       CFR          2      SYNC  86830     53.0      33.8    SYNC    19    0.2    4.5        2.7     14.5
            86.2
```

Figure 10-6 CF activity report CF-to-CF activity section sample

In addition, the new CF-to-CF section contains the information about the specific CF link types. This enhancement has also adapted to the existing subchannel activity section.

10.3 WLM report class periods support

Report classes allow you to report on any subset of the system's workloads regardless of the service class or classes they are associated with. A typical reporting goal would be, for example, to summarize by department. Since the concept of report class periods was missing, users tend to abuse their service class definitions for more reporting purposes. That is, the number of service classes with actually identical definitions is increased to achieve a higher level of reporting than provided by report classes.

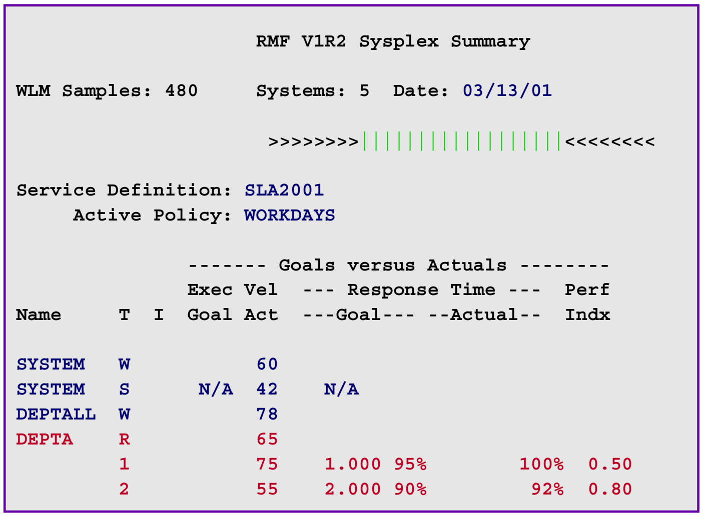
With z/OS V1R2, WLM provides reporting structures that meet customer's needs while keeping the number of service class definitions at a meaningful level. In detail, the following enhancements are introduced for WLM goal-mode report classes:

- Multiple periods
- Response time distribution
- Work manager delay status

RMF reporting is enhanced in the following ways:

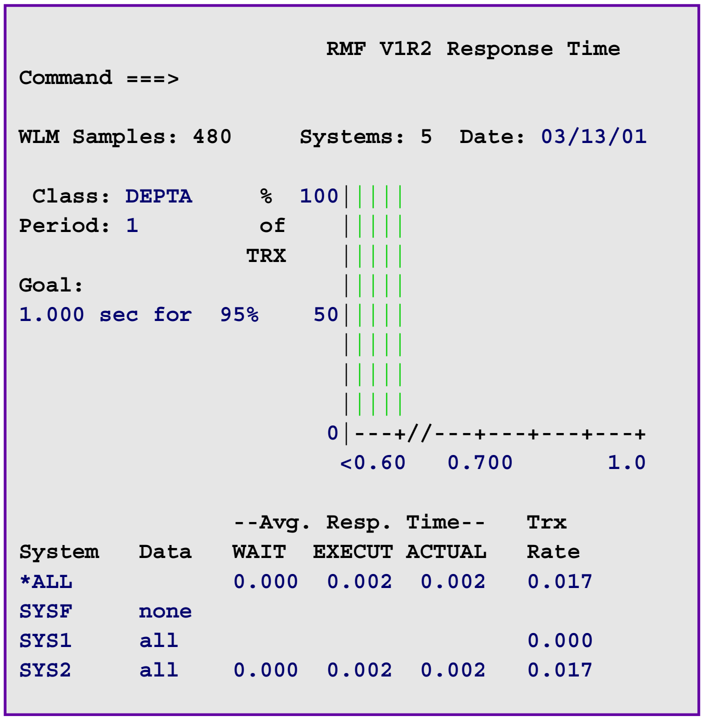
- All reports that formerly displayed one line per report class are now enabled to display multiple lines on report class period granularity.
- Those kinds of reports, which are related to one service class period, are now applicable in the same way for report class periods.

See Figure 10-7 and Figure 10-8.



```
                    RMF V1R2 Sysplex Summary

WLM Samples: 480     Systems: 5  Date: 03/13/01

                     >>>>>>>>|||||||||||||||||<<<<<<<<

Service Definition: SLA2001
     Active Policy: WORKDAYS

               ------- Goals versus Actuals --------
               Exec Vel  --- Response Time ---  Perf
Name     T  I  Goal Act  ---Goal--- --Actual--  Indx

SYSTEM   W           60
SYSTEM   S      N/A  42    N/A
DEPTALL  W           78
DEPTA    R           65
         1           75    1.000 95%       100%  0.50
         2           55    2.000 90%        92%  0.80
```

Figure 10-7 RMF Monitor III sysplex summary report sample



```
                       RMF V1R2 Response Time
Command ===>

WLM Samples: 480     Systems: 5  Date: 03/13/01

 Class: DEPTA      %  100|||||
Period: 1          of    |||||
                  TRX    |||||
Goal:                    |||||
1.000 sec for  95%     50|||||
                         |||||
                         |||||
                         |||||
                        0|---+//---+---+---+---+
                          <0.60   0.700      1.0

                --Avg. Resp. Time--   Trx
System   Data   WAIT  EXECUT ACTUAL   Rate
*ALL            0.000  0.002  0.002   0.017
SYSF     none
SYS1     all                          0.000
SYS2     all    0.000  0.002  0.002   0.017
```

Figure 10-8 RMF Monitor III response time distribution report sample

In contrast to service classes, periods are not explicitly defined for report classes. The number of periods is derived from a service class directly. Multiple service classes can contribute data to the same report class. As illustrated in Figure 10-9, the following relationships exist:

Heterogeneous relationship Customers, for example, can classify TSO users by accounting information (AI) and assign service classes DEPTA and DEPTB. DEPTA has different goals than DEPTB. However, for each TSO user in department DEPTA and DEPTB, only one report class TSOPROD is assigned.

Homogeneous relationship Customers can classify all TSO users to run in service class TSOPROD and distinguish the departments for reporting purposes by defining DEPTA and DEPTB respectively.

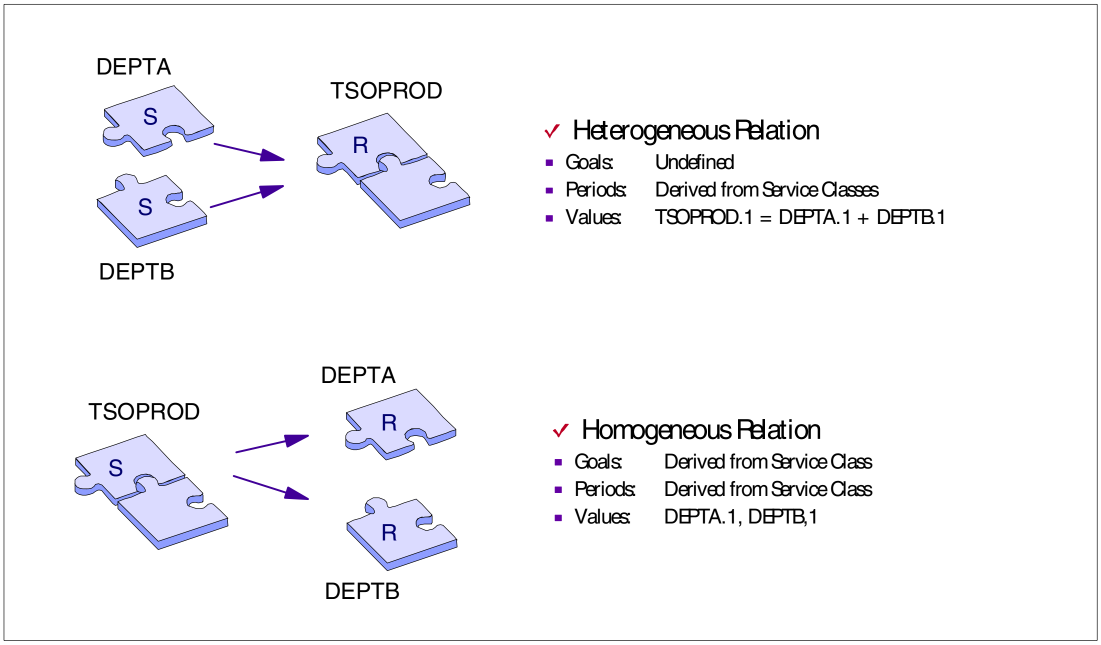

DEPTA
S
S
DEPTB
TSOPROD
R
✓ Heterogeneous Relation
▪ Goals: Undefined
▪ Periods: Derived from Service Classes
▪ Values: TSOPROD.1 = DEPTA.1 + DEPTB.1
TSOPROD
S
DEPTA
R
R
DEPTB
✓ Homogeneous Relation
▪ Goals: Derived from Service Class
▪ Periods: Derived from Service Class
▪ Values: DEPTA.1, DEPTB,1

Figure 10-9 Heterogeneous and homogeneous relationships

When more than one service class contributes to the same report class (heterogeneous relationship), the possibility exists that the same period in each service class has a different goal, importance, or duration. In addition, it is possible that the service classes contain different number of periods so that multi-period report classes might consist of data based on different conditions.

Since no well-defined goals are applicable for heterogeneous report classes, the response time distribution section, as well as the subsystem section, is omitted in the RMF Monitor I workload activity report.

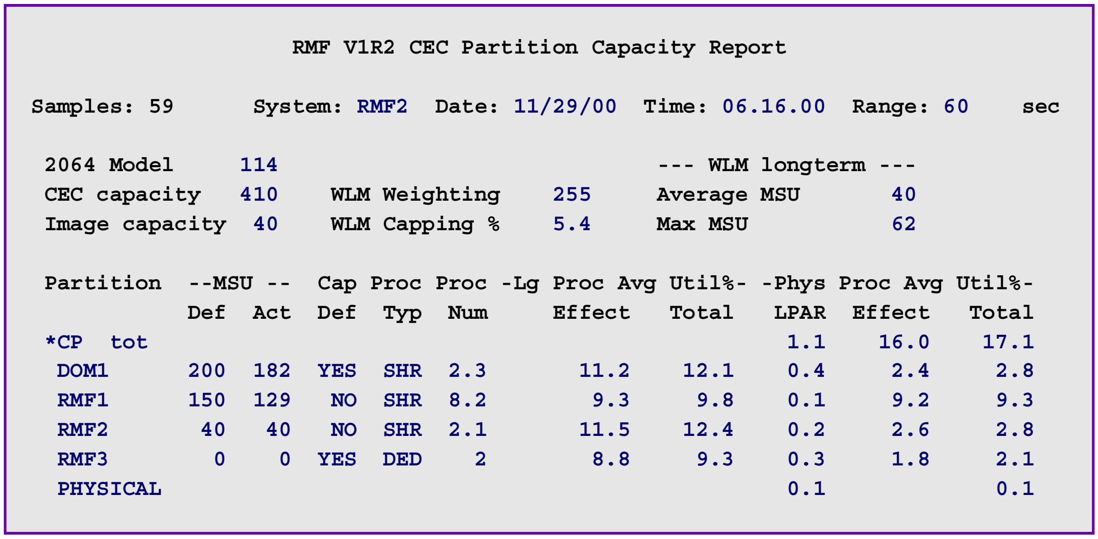
10.4 License manager support

Together with the new zSeries, the IBM License Manager (ILM) provides the ability to specify a defined capacity for each LPAR within one CEC. The sum of all capacity limits for the individual products running in the affected partitions is equal to the Product Certificate Capacity, which is taken as the base for the license charge. The ILM has the ability to handle workload spikes required by high priority workloads. Thus, one LPAR can consume more

MSUs than the specified capacity limit for a certain time period. Until the long term average value exceeds the limit, the partition is not capped by WLM. Thereby a partition cannot be capped under the defined million service units (MSU) limit. This ensures that the partition can still handle a basic amount of work.

10.4.1 RMF Monitor III

The Monitor III CEC Partition Capacity Report allows the user to keep track of whether the capacity definitions defined by the ILM are meeting the requirements for the workloads which are executing in the local partition. See Figure 10-10. An actual MSU consumption that comes close to the defined capacity can be taken as an indication to review the MSU limit definitions. This applies also for a high capping percentage. Due to the fact that the report is always refreshed with actual data, the user has the possibility to increase the limits dynamically in case the highest priority workloads are affected.



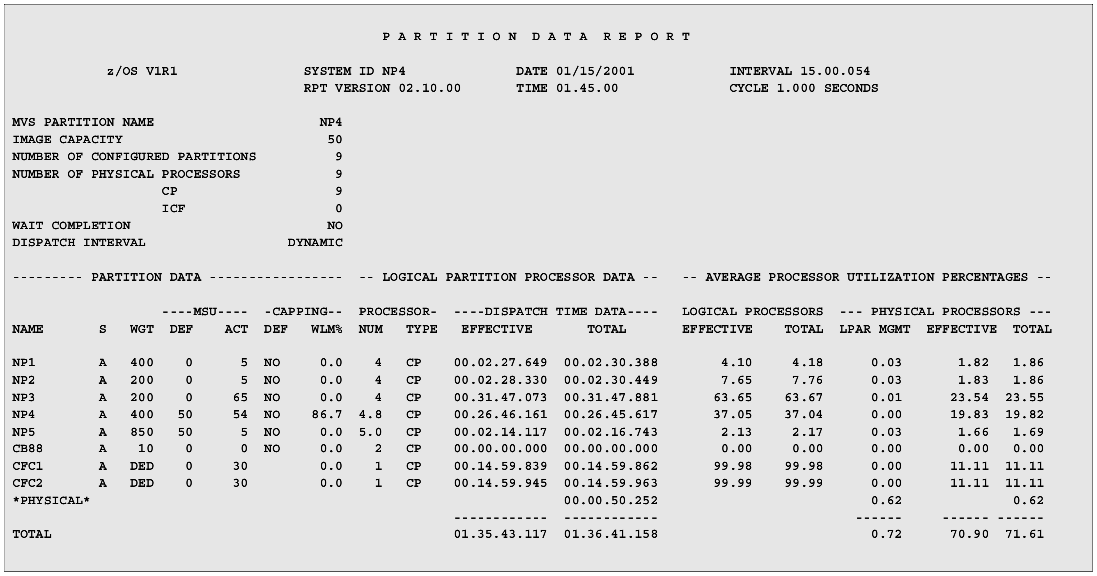
```
                    RMF V1R2 CEC Partition Capacity Report

Samples: 59      System: RMF2  Date: 11/29/00  Time: 06.16.00  Range: 60    sec

 2064 Model     114                             --- WLM longterm ---
 CEC capacity   410    WLM Weighting    255     Average MSU       40
 Image capacity  40    WLM Capping %    5.4     Max MSU           62

 Partition  --MSU --  Cap Proc Proc -Lg Proc Avg Util%- -Phys Proc Avg Util%-
            Def  Act  Def  Typ  Num     Effect   Total   LPAR  Effect   Total
 *CP  tot                                                 1.1    16.0    17.1
  DOM1      200  182  YES  SHR  2.3       11.2    12.1    0.4     2.4     2.8
  RMF1      150  129   NO  SHR  8.2        9.3     9.8    0.1     9.2     9.3
  RMF2       40   40   NO  SHR  2.1       11.5    12.4    0.2     2.6     2.8
  RMF3        0    0  YES  DED    2        8.8     9.3    0.3     1.8     2.1
  PHYSICAL                                                0.1             0.1
```

Figure 10-10 RMF Monitor III CEC partition capacity report

The bottom part of the report is derived from the existing Monitor I Partition Data Report and provides a CEC-wide overview about the CPU utilization from the logical as well as from the physical CPU perspective.

10.4.2 RMF Monitor I/Postprocessor

The RMF Partition Data Report has been extended to enable the users to keep track of the CPU resource consumption of the individual partitions. A sample is shown in Figure 10-11 on page 197.



```
                                               P A R T I T I O N   D A T A   R E P O R T

            z/OS V1R1                SYSTEM ID NP4              DATE 01/15/2001            INTERVAL 15.00.054
                                     RPT VERSION 02.10.00       TIME 01.45.00              CYCLE 1.000 SECONDS

MVS PARTITION NAME                     NP4
IMAGE CAPACITY                          50
NUMBER OF CONFIGURED PARTITIONS          9
NUMBER OF PHYSICAL PROCESSORS            9
                   CP                    9
                   ICF                   0
WAIT COMPLETION                         NO
DISPATCH INTERVAL                  DYNAMIC

--------- PARTITION DATA -----------------  -- LOGICAL PARTITION PROCESSOR DATA --   -- AVERAGE PROCESSOR UTILIZATION PERCENTAGES --

                   ----MSU----  -CAPPING--  PROCESSOR-  ----DISPATCH TIME DATA----   LOGICAL PROCESSORS  --- PHYSICAL PROCESSORS ---
NAME        S  WGT  DEF    ACT  DEF   WLM%  NUM   TYPE   EFFECTIVE       TOTAL       EFFECTIVE    TOTAL  LPAR MGMT  EFFECTIVE  TOTAL

NP1         A  400    0      5  NO     0.0    4   CP    00.02.27.649  00.02.30.388        4.10     4.18       0.03       1.82   1.86
NP2         A  200    0      5  NO     0.0    4   CP    00.02.28.330  00.02.30.449        7.65     7.76       0.03       1.83   1.86
NP3         A  200    0     65  NO     0.0    4   CP    00.31.47.073  00.31.47.881       63.65    63.67       0.01      23.54  23.55
NP4         A  400   50     54  NO    86.7  4.8   CP    00.26.46.161  00.26.45.617       37.05    37.04       0.00      19.83  19.82
NP5         A  850   50      5  NO     0.0  5.0   CP    00.02.14.117  00.02.16.743        2.13     2.17       0.03       1.66   1.69
CB88        A   10    0      0  NO     0.0    2   CP    00.00.00.000  00.00.00.000        0.00     0.00       0.00       0.00   0.00
CFC1        A  DED    0     30         0.0    1   CP    00.14.59.839  00.14.59.862       99.98    99.98       0.00      11.11  11.11
CFC2        A  DED    0     30         0.0    1   CP    00.14.59.945  00.14.59.963       99.99    99.99       0.00      11.11  11.11
*PHYSICAL*                                                            00.00.50.252                            0.62              0.62
                                                        ------------  ------------                          ------     ------ ------
TOTAL                                                   01.35.43.117  01.36.41.158                            0.72      70.90  71.61
```

Figure 10-11   RMF partition data report sample

New fields can be taken as the base for necessary MSU adjustments:

- The image capacity is the CPU capacity available to the z/OS image measured in MSUs per hour. In case no limit has been specified explicitly, the capacity derived from the logical CP configuration of the LPAR is displayed.
- The defined and consumed MSUs are reported. A defined value of 0 means that softcap is not applicable for this partition and the full capacity is taken as the License Managers basis for the charge.
- CAPPING WLM% is the percentage of time where WLM has capped the partition. This happens if a partition has exceeded its capacity limit for a certain time period (soft capping).

In addition to the report extensions, the following new overview criteria are introduced:

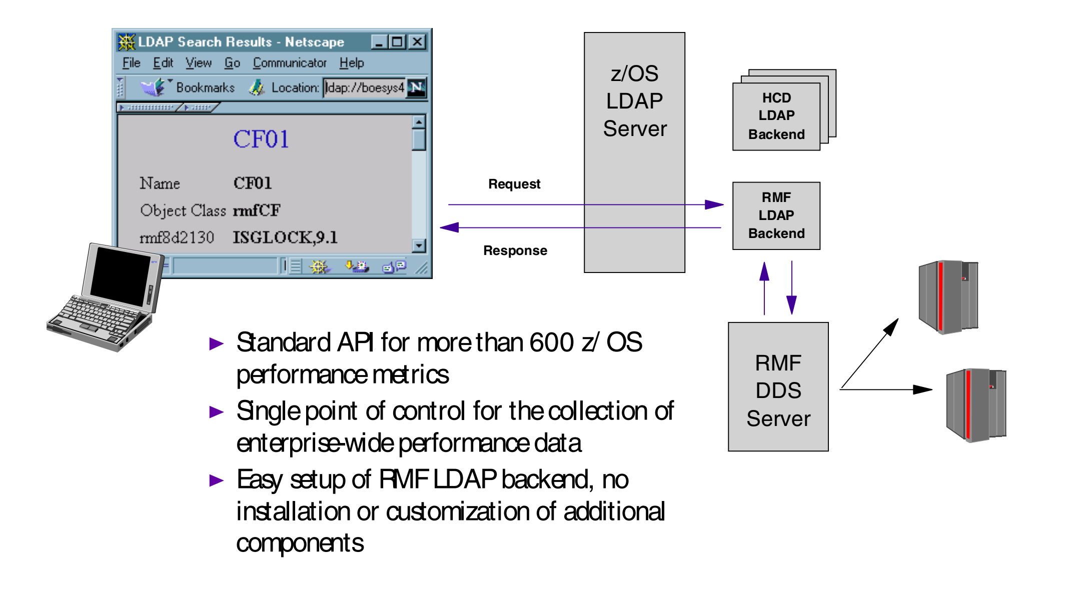
LDEFMSU Defined capacity limit in units of MSU

LACTMSU Actual number of consumed MSUs

WCAPPER Percentage of WLM capped in the partition

10.5 LDAP integration

RMF performance data will now be accessible by an open and standardized interface, the Lightweight Directory Access Protocol (LDAP). The RMF LDAP directory is like a database, but without its own persistent storage. LDAP provides well-defined access methods to structured data. Using LDAP, other performance and systems management components have easy access to performance-relevant data within a z/OS environment. The RMF LDAP backend routes the incoming requests to the RMF Distributed Data Server (DDS). Thus, more than 600 performance data can be retrieved by means of simple LDAP query requests. See Figure 10-12 on page 198.


LDAP Search Results - Netscape
File Edit View Go Communicator Help
Bookmarks Location: ldap://boesys4
z/OS
LDAP
Server
HCD
LDAP
Backend
Request
Response
RMF
LDAP
Backend
RMF
DDS
Server

CF01

Name CF01

Object Class rmfCF

rmf8d2130 ISGLOCK,9.1

- Standard API for more than 600 z/ OS performance metrics
- Single point of control for the collection of enterprise-wide performance data
- Easy setup of RMF LDAP backend, no installation or customization of additional components

Figure 10-12 RMF LDAP integration

10.6 3270 local session removal

With the removal of BTAM, the RMF Monitor II local 3270 session is no longer supported. With the introduction of the Monitor II ISPF enabling, all reports can be accessed by means of the ISPF workstation.

Even though the local session was rarely used, it was a vehicle to obtain RMF monitor performance data without an active TSO subsystem. However, you can have access to Monitor II reports without an active TSO/TCAS subsystem by means of the RMF Client/Server Enabling (RMFCS). We should note that the workstation software only supports the OS/2 operating system.

11

JES3 enhancements

In this chapter, changes to JES3 Version 1 Release 2 are described. These enhancements allow you to:

- Define job numbers up to 999999
- Reduce the job number range with a hot start with refresh and no IPL

In addition, migration considerations are discussed.

11.1 JES3 job number support

The OPTIONS initialization statement now allows for a larger maximum amount of job numbers. This allows users to select a job number range that should not have the same job number in the system in the same day or several days. The new highest job number can now be 999999.

The JOBNO parameter on the OPTIONS statement is specified as follows:

```
JOBNO=(lowest,highest,joblim)
```

where:

lowest Specifies a number within the range 1 to 999999 to indicate the lowest job number.

highest Specifies a number within the range 18 to 999999 to indicate the highest job number.

joblim Specifies the maximum number of jobs that may be in the JES3 complex at any given time (within the range 1 to 999999).

The JOBNO parameter on the OPTIONS statement can now be defined as follows:

```
JOBNO=(1,999999,999999)
```

There are other limits in JES3 that are not changed in this release, as follows:

- The JSPAN keyword on GROUP still has a maximum of 65,534.
- 65,534 active JES-managed jobs per processor.
- 65,534 active WLM-managed jobs per processor.
- 32,767 jobs per DJC net.
- 9,999 BDT jobs and job numbers.

Note: Job numbers and job counts that reside in various data areas in JES3 control blocks are changed from 2-byte fields to 4-byte fields. In some cases they will not be fullword aligned, so the ICM instruction must be used instead of the L instruction in any user code.

11.1.1 New jobids

When you specify a job number range greater than 99999 on the OPTIONS statement in the JES3 initialization stream, jobids are assigned with the format of Jnnnnnnn. Jobids that are greater than 99999 are now displayed in messages with the prefix of J instead of JOB, as follows:

- For jobids with a job number of 99999 or less, messages are shown as follows:

  ```
  IAT8674 JOB JOBABC (JOB17437) P=15 CL=A       OUTSERV(PENDING WTR)
  ```

- For jobids with a job number greater than 99999, messages are shown as follows:

  ```
  IAT8674 JOB JOBXYZ (J0117437) P=15 CL=A       OUTSERV(PENDING WTR)
  ```

The six digit job number is left padded with 0 to make it seven digits. If the operator types in a seven digit number in a JES3 command such as **`*I J=1234567`**, the error message that appears is as follows:

```
IAT8610 JOB (J1234567) NOT FOUND
```

New formats

When the job number is greater than 99999, the following are the new formats:

- J0394827 for batch jobs
- S0593847 for started tasks
- T0384723 for TSO logons

> **Note:** JES3 will not build jobids of the form Jxxxxxxx for any job from 1-99999 even if the maximum is specified to be greater than 99999 in the initialization stream.

Initiator jobid format

In some cases initiator jobids are built as INTxxxxx instead of JOBxxxxx. When the initiator job number is greater than 99999, the jobid is changed to Ixxxxxxx.

SMF type 26 records

For SMF26 records only, jobids are built in the following formats:

- TSUxxxxx for TSO logons
- STCxxxxx for started tasks

When you specify a job number range greater than 99999 on the OPTIONS statement, the following formats are used:

- Txxxxxxx for TSO logons
- Sxxxxxxx for started tasks

NJE jobids

NJEROUT jobs are assigned a new special jobid of RRTxxxxx. This jobid can be used an alternate name for the DSP when the operator issues `*S` and `*C` commands. If the NJEROUT job number is assigned as 1837, it is displayed in a message as RRT1837. The operator specifies as follows:

- **`*S RRT1837`** or
- **`*C RRT1837`** for this FCT

If the jobid and alternate FCT name have a number greater than 99999, the jobid is displayed as Rxxxxxxx.

11.2 Hot start with refresh

The real job number range that JES3 supported in previous releases was 65534 due to the fact that a half-word was used to store the number and signed arithmetic instructions were used to access the fields. Coexistence and fallback to previous releases from JES3 1.2 is allowed, but jobs greater than 65534 cannot be defined until the entire complex is at the JES3 1.2 level, and fallback is not possible once jobs greater than 65534 are defined.

The JOBNO range on the OPTIONS statement can be expanded during a hot start with refresh, but the reverse is not true on all previous releases of JES3. To lower or shrink the range requires a warm start with all previous releases. This was a restriction introduced in OS/390 JES3 2.4, which introduced both the hot start with refresh and job numbers greater than 32767.

With z/OS JES3 1.2, the restriction is removed. The job number range will be allowed to be shrunk during a hot start with refresh. However, JES3 will make no attempt to stop an existing running job. If you change the job number range by lowering the range, you are asked by replying to a WTOR if you want to delete jobs that are now outside the new range.

Note: If you change the JOBNO parameter during a hot start with refresh such that jobs are deleted, and no IPL is being performed, it is your responsibility to make sure that no jobs that are being deleted are active on main. JES3 issues messages IAT4131 and IAT4133 for all jobs that are deleted by a change in the JOBNO parameter. You can ignore these messages for active jobs provided that for every active job that is deleted, the processor on which the job was running is IPLed.

Because of this ability to shrink the job number range, the following steps should be taken:

- Active jobs outside the range must be canceled.
- The processor where these jobs are running must be IPLed.

  If you do not IPL the processor on which jobs to be deleted are active, you must either wait for these active jobs to end or cancel them. The active jobs must end before you perform this hot start with refresh.

Note: If you lower the upper end of the job number range or increase the lower end of the job number range, jobs that are in the system and were previously assigned job numbers outside the new range are lost.

The way to avoid losing jobs when you change the upper end of the range is to do a warm start. However, before making the change, execute the dump job (DJ) facility to save jobs that are in the job queue. After the change, again execute the dump job facility to restore these jobs to the job queue.

11.3 Long running jobs support

Long running jobs such as VTAM or HSM generate many messages that are written into the JESMSGLG and JESYSMSG data sets. These data sets reside on the JES3 spool until the long running job ends. This can cause the spool to contain data that is not needed and may affect the spool capacity. The objective of this support is to implement a way to spin off these data sets prior to the long running job ending. The JESMSGLG and JESYSMSG data sets are often referred to as the JESlog data sets. The messages that are created during the execution of a job are copied to the JESMSGLG data set while the job is executing. If the job runs for a long time, which can be for weeks, then the SPOOL usage for these messages could create a problem for some installations.

New externals are added to allow an installation to control which jobs should have the JESlog data sets spun off for print processing prior to the job completing execution and, more importantly, to save spool space. Installations can mark a job eligible for JESlog spin processing by specifying:

- JESLOG= on a job's JOB card JCL statement
  - The specification of the JOB statement applies to just that specific job. The specification on the JOB statement overrides any default specifications on the CLASS statement.
- SPIN= on the CLASS statement in the JES3 initialization stream
  - All jobs running in this class will have the SPIN= attribute applied to them.

- The operator **`*F J=xxxxx,SPIN`** command for those jobs identified as needing to spinoff the JESlog data sets

> **Note:** When the spin-off limit value is exceeded, both JESlog data sets are spun off.

11.3.1 JOB JCL statement

The JESLOG= keyword on the JOB JCL statement supports the following parameters:

- JESLOG=SPIN or JESLOG=(SPIN)
  - Spin off when the operator issues the **`*F J=`** command
- JESLOG=(SPIN,'+hh:mm')
  - Spin off (SPIN,+03.00) at every 3 hours interval

    > **Note:** If the spin-off interval time is set to occur every 3 hours and the last spin-off occurred at 3:00 PM and the operator issues a *F J= command at 4:00 p.m. to spin-off the JESlog data sets, the next scheduled spin-off time is now set to 7:00 PM.

- JESLOG=(SPIN,'hh:mm')
  - Spin off (SPIN,12:30) at a specific time 12:30 PM

    > **Note:** If the spin-off TOD time is set for 12:30 PM, and the job begins executing at 12:31 PM, spin-off occurs at 12:30 PM the next day.

- JESLOG=(SPIN,nnn)
  - Spin off (SPIN,5000) every 5000 lines
- JESLOG=(SPIN,nnnK)
  - Spin off (SPIN, 5K) every 5 thousand lines
- JESLOG=(SPIN,nnnM)
  - Spin off (SPIN,2M) every 2 million lines
- JESLOG=SUPPRESS or JESLOG=(SUPPRESS)
  - No messages are written to the JESlog data sets during job execution.
- JESLOG=NOSPIN or JESLOG=(NOSPIN)
  - The NOSPIN option on the JOB statement can be used to override any specification on the CLASS statement to make sure no messages are written to the JESlog data sets.

11.3.2 CLASS initialization statement

A new keyword on the CLASS initialization statement, SPIN=, is added to allow an installation to allow or suppress spinoff of JESlog data sets during job execution. The SPIN keyword on the CLASS initialization statement supports the following parameters:

```
CLASS,...SPIN=<YES|+hh:mm|hh:mm|nnn|nnnK|nnnM|NO>
   DEFAULT: NO
```

As shown in Table 11-1 on page 204, SPIN=YES is equivalent to JESLOG=SPIN on the JCL statement and SPIN=NO is equivalent to JESLOG=NOSPIN.

The other options for SPIN= on the CLASS statement are equivalent to the specifications on the JCL statement, as discussed in “JOB JCL statement” on page 203.

JESMSG keyword

The CLASS statement has always had a JESMSG=LOG|NOLOG keyword and parameter. The JESMSG keyword specifies whether to allow (LOG) or suppress (NOLOG) writing WTOs and WTORs to the JESMSGLG data set for jobs in this job class. Since this keyword can also be specified together with the SPIN= keyword, the following considerations are necessary.

> **Note:** If the same CLASS statement is defined with JESMSG=NOLOG and SPIN= anything other than NO, JES3 issues a warning message and assumes SPIN=NO is to be used.

Operator class command

You can now use the **`*F C=class`** command to:

- Control the spinning of JESlog data sets
- Control writing to the JESMSGLG data set

The new keywords supported are as follows:

JESMSG=LOG|NOLOG|STD

SPIN=NO|YES|hh:mm|+hh:mm|nnn|nnnK|nnnM

By specifying these keywords, you can change or add the spin options and the JESMSG option dynamically, as shown in the following example:

```
*F C=A,SPIN=YES
```

11.3.3 Parameter comparison

Since there are two ways to implement the JESlog data set spinoff, Table 11-1 shows a comparison between the JCL statement and the CLASS statement as to how they implement the same function.

Table 11-1 SPIN keyword in JCL and initialization stream comparison

JOB JCL statement	CLASS initialization statement
JESLOG=SPIN	SPIN=YES
JESLOG=NOSPIN	SPIN=NO
JESLOG=(SPIN,'+hh:mm')	SPIN=+hh:mm
JESLOG=(SPIN,'hh:mm')	SPIN=hh:mm
JESLOG=(SPIN,nnn)	SPIN=nnn
JESLOG=(SPIN,nnnK)	SPIN=nnnK
JESLOG=(SPIN,nnnM)	SPIN=nnnM
JESLOG=SUPPRESS	JESMSG=NOLOG

11.3.4 Operator command for spinoff

A new keyword, SPIN, is supported on the `*F J=` command. This command allows for the JESlog data sets to be spun off, if made eligible by the SPIN= keyword on the CLASS initialization statement or the SPIN keyword on the JOB JCL statement, as shown in Figure 11-1 on page 204.

Two new options have been added to the `*F J=` command as follows:

SPIN Requests that the JESlog data sets be spun off for the specified job

MSG= Specifies text of up to 120 bytes to be added from the operator console to the job's JESMSGLG data set

11.3.5 New JESMSGLG messages

JES3 provides new messages into the JESMSGLG data set. The first message, IAT6853, appears at the top of the data set as follows:

```
 IAT6140 JOB ORIGIN FROM GROUP=ANYLOCAL, DSP=IR , DEVICE=INTRDR  , 0000
 08:38:25 IAT6853 THE CURRENT DATE IS SUNDAY,    27 JAN 2002
IRR010I  USERID VAINI    IS ASSIGNED TO THIS JOB.
 08:38:25 IAT4802 ATTEMPTED DEBUG OPTIONS ARE AS FOLLOWS
 08:38:25 DEBUG=ALL
 08:38:25 IAT4401  LOCATE FOR STEP=E1       DD=SYSUT1   DSN=SYS1.PROCLIB
 08:38:25 IAT4402 UNIT=3390    ,VOL(S)=TOTSY1
```

Figure 11-1 New message in JESMSGLG data set

The second message, which is applicable only to those jobs that are spin eligible, IAT6819, appears at the end of the JESMSGLG data set to indicate the reason why the spinoff occurred. The syntax of this message is as follows:

```
{JESMSGLG}|{JESYSMSG} DATASET SPINOFF DUE TO {OPERATOR COMMAND}| {TIME OF DAY LIMIT
REACHED}|{TIME INTERVAL REACHED}|{LINE LIMIT REACHED}|{END OF JOB}
```

For example:

```
IAT6819 JESMSGLG DATASET SPINOFF DUE TO TIME OF DAY LIMIT REACHED
```

11.4 Migration considerations

APAR OW47435 is the compatibility APAR for z/OS JES3 1.2 (HJS7705). This APAR prevents a customer from inadvertently falling back to a release prior to HJS7705 over a hot start with refresh if they have started to make use of HJS7705 functions. This APAR will handle a situation on a fallback to a previous release if the new system has defined a job number range on the OPTIONS initialization statement with a high job number greater than 65534, or submitted a job or started a task with the spinoff attribute, and that job is still in the job queue during the restart. In addition, this APAR allows the customer to fall back to the previous release with a hot start with refresh and reduce the job number range at the same time. Falling back could mean re-IPLing the global with a lower release level, or performing a DSI that places the new global on a lower release level.

APAR OW47435 enforces the following OS/390 Releases of JES3 to coexist with z/OS JES3 1.2:

- JES3 2.8
- JES3 2.9

- JES3 2.10

APAR OW47435 allows OS/390 Releases JES3 2.8, JES3 2.9, and JES3 2.10 to recognize and react to the attempted use of z/OS JES3 1.2 functions while falling back. Without safeguards during initialization, various problems can occur, such as:

- An ABENDDM747 in module IATDMJA.
- MSGIAT4174 issued with an incorrect jobid.
- A truncation of the job number range, which leads to an immediate loss of jobs with the following messages issued: MSGIAT4174, MSGIAT4133, and/or MSGIAT4134.
- Failure of subsequent job submission or started tasks with MSGIAT6340 issued.
- Premature detection of a job number shortage with MSGIAT6341 issued.

This APAR is required to do the following:

- Allows a fallback to reduce the job number range at the same time and do a hot start with refresh.
- Deletes active jobs that are using JESLOG=SPIN.
- Recognizes a job that is dumped using Dump Job (DJ) that has a job number greater than 65534 and assigns it a new number.

11.4.1 Coexistence restriction

z/OS JES3 1.2 can coexist with OS/390 JES3 Releases 2.8, 2.9, and 2.10, or z/OS 1.1. The following restriction apply:

- The entire complex must be z/OS JES3 1.2 before you can increase the JOBNO range beyond 65534.
- JES3 does not allow a local below z/OS JES3 1.2 to connect to a global that has job numbers greater than 65534.

11.4.2 z/OS JES3 1.2 enablement

The long running job function has been disabled in JES3 1.2 with the following APAR:

- OW51878

The APAR that will enable this function is:

- OW50563

Check the status of these APARs before activating this function.

12

Language Environment

This chapter describes the enhancements that are introduced with z/OS Version 1 Release 2 Language Environment Favour 31, as follows:

- Favour 31 installation
- Language Environment options
- Migration to Release 2
- Behavior of applications

12.1 Favour 31 overview

The strategic enhancement introduced with this release of z/OS is to reduce overall initialization/termination path length, while virtually eliminating below-the-line storage, for 31-bit applications. Favour 31 applies only to HLL applications run under z/OS Language Environment Release 2.

This enhancement addresses concerns regarding performance and below-the-line storage usage when migrating applications, and the inability for new applications, for example, multi-process and multi-threaded, to reach an optimum scale. It has been indicated that pre-Language Environment run-time environments are better performing and use less storage, below-the-line and above-the-line.

Language Environment will assume a 31-bit application, rather than worrying that there might be a 24-bit routine invoked somewhere during execution. With this assumption, Language Environment can optimize storage usage and initialization path length.

Note: PL/I applications continue to require below-the-line storage.

Applications are now required to explicitly tell Language Environment that a 24-bit routine might be called. This is a 180-degree reversal from previous default operation (non-CICS).

This is why it is called “Favour 31.”

Note: The ALL31 run-time option is one way for an application to tell Language Environment whether or not a 24-bit routine might be called. Another method is for the “main” program to be AMODE 24.

12.2 Favour 31 invocation

z/OS 1.2 Language Environment assumes that applications will be 31-bit. It will allocate storage under that assumption, and react when an AMODE 24 main or the ALL31(OFF) option is found.

Customers must be aware of the migration concerns.There is no mechanism to tell z/OS 1.2 Language Environment not to assume a 31-bit application other than to change CEEDOPT defaults back to OS/390 2.10 values.

12.3 Favour 31 installation

All HLL applications that use Language Environment on z/OS 1.2 should be tested before putting them into production. This should flush out those applications that cannot run with the new run-time option defaults.

To ensure that applications continue to execute without being affected by the changes introduced with Favour 31, one method would be for the System programmer to change Language Environment run-time option defaults (CEEDOPT) for ALL31 and STACK back to OS/390 2.10 values.

An alternate method would require that the application programmer could update those applications that invoke AMODE 24 routines to include ALL31(OFF) and STACK(,,BELOW) settings in a CEEUOPT.

> **Important:** The recommendation is for the defaults to be set at the OS/390 2.10 values and for the CEEUOPT to be created and used as the reverse to the default options set within CEEDOPT.

12.4 Language Environment options

The default run-time options for z/OS 1.2 Language Environment come with changes to the installation-wide default (CEEDOPT) run-time options (non-CICS). In Figure 12-1 we highlight the default options that have changed in Release 2.

ALL31=((**ON**),OVR)

STACK=((128K,128K,**ANYWHERE**,KEEP,512K,128K),OVR)

STORAGE=((NONE,NONE,NONE,**0K**),OVR)

THREADSTACK=((OFF,4K,4K,**ANYWHERE**,KEEP,128K,128K,OVR)

Figure 12-1 z/OS 1.2 Language Environment defaults

These new default options *will* affect existing applications that invoke 24-bit routines, or otherwise rely on a 24-bit addressable stack. The ALL31 run-time option is what really controls whether or not Favour 31 is “active.”

When the initial program is AMODE 31 and ALL31(OFF) is *not* found during options merge, then Favour 31 is considered to be active.

The recommendation is to revert back to OS/390 2.10 defaults (CEEDOPT) for these options, as this would allow for the applications to continue to process. The OS/390 Language Environment defaults are illustrated in Figure 12-2.

ALL31=((**OFF**),OVR)

STACK=((128K,128K,**BELOW**,KEEP,512K,128K),OVR)

STORAGE=((NONE,NONE,NONE,**8K**),OVR)

THREADSTACK=((OFF,4K,4K,**BELOW**,KEEP,128K,128K),OVR)

Figure 12-2 OS/390 Language Environment defaults

With ALL31(ON) and STACK(,,ANY) as the default options, the result is a reduction in the path length during initialization. This eliminates several page faults and has reduced the number of GETMAIN requests. This is only relevant to the non-CICS environment, since the default for CICS options (CEECOPT) has been in place for several years.

There are several other areas where below-the-line storage has resulted in a storage reduction. These are as follows:

- Static Library Vectors (saved approximately 25KB above the line).
- Control bock reorganization.
- Deferred LIBSTACK allocation.

  - Several language environment routines were updated to no longer use the LIBSTACK for their processing.
- Reserve stack size default is 0K.
  - The reserve stack is now allocated with the STACK (THREADSTACK) storage. The location of the STACK now determines the location of the reserve stack.
- Eliminated unused storage.
  - One of the larger storage reductions was the removal of the CICS specific block storage from the non-CICS environments.
- Static Library Vectors with OS/390 2.10 (and earlier): Language Environment code behind the external interfaces (AWIs) and vendor interfaces (CWIs) are accessed through two library vectors that are dynamically built during initialization. The vectors were required to be dynamic since multiple load modules shared the same vector. The vectors were updated as the modules were loaded.
  - Merged load modules

    CEEOLVD, CEEQMATH and CEELCLE have been merged into CEEPLPKA.
  - Saves approximately 25KB above-the-line storage
  - Saves a GETMAIN request
  - Reduces initialization path length

Note: Because of the module merge, CICS ERDSA increase of approximately 200K-300K is required.

The elimination of CEEQMATH caused problems during CICS region (partition) initialization. CICS APARs are available to allow z/OS 1.2 to operate under the supported CICS releases.

The initialization Control Block Footprint (non-CICS) storage savings as shown in Table 14-1 are approximate values. When Favour 31 is active, the below-the-line control blocks are actually allocated above.

Table 12-1 Control block footprint example

Release	Above CB Size	Below CB Size	GETMAIN #
OS/390 2.8	99148(13112)	14400(808)	6
OS/390 2.9	99148(13112)	14400(808)	6
OS/390 2.10	100956(13792)	14656(808)	7
z/OS 1.2	59496(13792+16)*	0	2

(*) The above-the-line storage allocation includes the traditional below-the -line control blocks when Favour 31 is active.

Control block sizes do not include C/C++, COBOL, PL/I or FORTRAN run-time library-specific control blocks. Stack and heap storage are not part of control block storage. Stack and heap storage can be tuned.

12.5 Migration considerations

The following applications and environments are affected:

- Non-CICS applications that begin with an AMODE 31 main and invoke at least one AMODE 24 routine during execution. If the environment had been initialized with the new defaults of ALL31(ON) and STACK(,,ANY), the application will most likely no longer work.
- PIPI adds an entry of an AMODE 24 routine into a PIPI subroutine environment that is running with the new defaults of ALL31(ON) and STACK(,,ANY) will not be detected. Invocation of that routine will most likely cause an error, due to the Language Environment control blocks and STACK being above-the-line.

To resolve this, do the following:

- Link a CEEUOPT with ALL31(OFF), STACK(,,BELOW) with the main program in those applications that have been taking advantage of the old defaults.
- Specify ALL31(OFF), STACK(,,BELOW) on the PIPI initialization subroutine invocation when an AMODE 24 routine will be used.
- Have the system programmer restore CEEDOPT back to ALL31(OFF) and STACK(,,BELOW) until all applications can be updated.

12.6 Behavior of applications

When you set your initialization default run-time options, consider the following when running applications in z/OS Version 1 Release 2.

12.6.1 Batch applications

When running batch applications, consider the following:

- The assumption is that ALL31(ON) is in effect.
- Both the above-the-line and below-the-line control blocks are allocated together above-the-line, except when either of the following is true:
  - The main program is AMODE 24.
  - CEEUOPT specifies ALL31(OFF).
- ALL31(OFF) after options merge will cause:
  - True below-the-line control blocks copied below-the-line. **ABEND U4093-D4** will be issued if there is a problem acquiring the true below-the-line storage after ALL31(OFF) has been detected.
- Internal pointers are fixed up to refer to the new storage.
- Optimally, 0 bytes below-the-line after initialization.
- Includes UNIX System Services applications.

12.6.2 Library routine retention (LRR)

Language Environment library routine retention is a function that provides a performance improvement for applications or subsystems. The following should be considered:

- The assumption is that ALL31(ON) is in effect.
- Both the above-the-line and below-the-line control blocks are allocated *together* above-the-line.

- First AMODE 24 or ALL31(OFF) transaction will cause:
  - True below-the-line control blocks copied below-the-line. Once true below-the-line control blocks have been copied below, the LRR environment will remain in that mode until the environment is terminated.
  - Internal pointers are fixed up to refer to the new storage.

12.6.3 Pre-init compatibility interface (PICI)

- All storage continues to be below-the-line.
- Benefits from reduction of control block sizes.

12.6.4 CICS

There are no changes to CICS initialization, which had already been changed in previous releases to use as little below-the-line storage as possible. Path length improvements had also been made.

A new default under CICS is LIBSTACK=(32,4080,FREE). The initial and increment sizes for LIBSTACK are rounded to the next higher multiple of 8 bytes. The default increment size under CICS of 4080 bytes accommodates the 16 bytes CICS storage check zone. Without this accommodation, an extra page of storage is allocated (only when the storage allocation is below the 16 MB line).

12.6.5 PIPI main

Language Environment uses the PIPI table to identify the routines that are candidates for execution in the pre-initialized environment. The following conditions apply:

- The assumption is that ALL31(ON) will be in effect.
- Both the above-the-line and below-the-line control blocks are allocated together above-the-line, except when either of the following is true:
  - The caller of PIPI initialization main is AMODE 24.
  - At least one routine in the PIPI table is AMODE 24. This might not be determined until after routines are loaded.
- First AMODE 24 or ALL31(OFF) main will cause:
  - Once true below-the-line control blocks have been copied below, the PIPI main environment will remain in that mode until the environment is terminated.
  - If there is an AMODE 24 routine detected after the routines are loaded (by Language Environment), then the true below-the-line control blocks are copied below-the-line and the pointers are fixed up to refer to the new storage.
  - Optimally, there will be 0 bytes below-the-line after initialization.

PIPI subroutine

- The assumption is that ALL31(ON) will be in effect.
- Both the above-the-line and below-the-line control blocks are allocated together above-the-line, except when:
  - The caller of PIPI initialization subroutine is AMODE 24.
  - At least one routine in the PIPI table is AMODE 24. This might not be determined until after routines are loaded.

- ALL31(OFF) after options merge will cause:
  - True below-the-line control blocks are copied below-the-line.
  - Internal pointers are fixed up to refer to the new storage.
- Optimally, there will be 0 bytes below-the-line after initialization.

12.6.6 PL/I environment

When PL/I is brought into the environment, this causes an initialization/termination LIBSTACK (2KB) to be allocated which:

- Is required for PL/I termination
- Is always allocated below-the-line
- Applies to batch, LRR, and PIPI environments only

ABEND U4093-D8 will be issued if there is a problem acquiring the initialization/termination libstack when PL/I is brought into the environment.

ABEND U4094-54 will be issued if there is a problem releasing the initialization/termination libstack acquired due to PL/I being brought into the environment.

12.6.7 Stack and heap

Stack and heap allocation changes are as follows:

- The reserve stack, size specified in the 4th sub-option on the STORAGE run-time option, will be allocated where the stack is allocated.
- When ALL31(ON), STACK(,,ANY), ANYHEAP(,,ANY) and HEAP(,,ANY) are in effect, the reserve stack, the initial stack segment, and the initial segments for the two heaps are allocated together with a single GETMAIN request.
- The LIBSTACK allocation is deferred.

12.6.8 GETMAIN requests

GETMAIN requests when Favour 31 is active are as follows:

- Reduced from 7 to 2 for POSIX(OFF) C programs.
- Reduced from 7 to 3 for COBOL for MVS and VM programs.
- Reduced from 8 to 4 for PL/I for MVS and VM programs.

13

RACF enhancements

This chapter describes the enhancements to RACF available with z/OS Version 1 Release 2. The following topics are discussed:

- Group size expansion
- SAFTRACE
- Support of mixed case profile names
- Support for DB2 version 7
- Kerberos extensions

13.1 RACF

Access, in a computer-based environment, means the ability to do something with a computer resource (for example, use, change, or view something). Access control is the way by which this ability is explicitly enabled or restricted. Computer-based access controls are called logical access controls. Logical access controls are protection mechanisms that limit users' access to information and restrict their forms of access on the system to only what is appropriate for them. Logical access controls are often built into the operating system, or may be part of the logic of applications programs or major utilities, such as Database Management systems. They may also be implemented in add-on security packages that are installed into an operating system; such packages are available for a variety of systems, including PCs and mainframes. Additionally, logical access controls may be present in specialized components that regulate communications between computers and networks. The Resource Access Control Facility (RACF) controls access to all protected MVS resources. RACF protects resources by granting access *only* to authorized users of the protected resources. RACF retains information about the users, resources, and access authorities in profiles on the RACF database and refers to the profiles when deciding which users should be permitted access to protected system resources. For more detailed information about this, refer to *z/OS SecureWay Security Server RACF Security Administrator's Guide*, SA22-7683.

13.2 Group size expansion

The group structure of RACF can be mapped to the organizational structure that exists at your installation. That is, RACF conforms naturally to a tree structure of groups, where each group (except SYS1, which is predefined as the highest group) has a superior, or owning group. Groups can correspond directly to business entities such as divisions, departments, and projects. Users can be connected to one or more groups.

When you define a group, you should consider the basic purpose of the group. Is it an administrative group, a holding group, a data control group, a functional group, or a user group? When setting up RACF groups, keep in mind that the maximum number of users that you can connect to any one group is 5957.

Note: For more details about the types of groups (administrative, holding, data control, functional, and user), see *z/OS V1R2.0 SecureWay Security Server RACF Security Administrator's Guide*, SA22-7683.

For groups that may become large, with an unlimited number of users that have no special authority, and that are unlikely to be deleted, such as a group that contains all users, and for which a quick listing of members is not needed, you may wish to consider defining the groups using the UNIVERSAL operand of the ADDGROUP command. Universal groups may be appropriate for holding groups and other types of groups.

These users, whose numbers are unlimited, have a group authority of USE, and can not have any group-related user attributes (group-SPECIAL, group-OPERATIONS, group-AUDITOR). Universal groups may contain users with a level of authority higher than USE and/or group-related attributes, but their numbers are limited to the maximum number of users allowed in a non-UNIVERSAL group (5957 members).

Universal groups are user groups that do not have complete membership information stored in their group profiles. The benefit is that there is no limit on the number of group members. However, you cannot list all the members using the LISTGRP command.

The UNIVERSAL attribute can be defined for a group only at group creation time. The UNIVERSAL attribute cannot be added or removed using the ALTGROUP command. The output of the LISTGRP command now contains the UNIVERSAL attribute for universal groups, as shown in Figure 13-1 and Figure 13-2.

```
ADDUSER GENADMIN NAME(GENADMIN) OWNER(MARTINS) +
        UACC(READ) DFLTGRP(SYS1) AUTHORITY(USE)
ADDGROUP (GENUSERS) OWNER(GENADMIN) UNIVERSAL
```

Figure 13-1 Commands to add user GENADMIN and add group GENUSERS

```
LISTGRP GENUSERS
INFORMATION FOR GROUP GENUSERS
    SUPERIOR GROUP=SYS1         OWNER=GENADMIN
    NO INSTALLATION DATA
    NO MODEL DATA SET
    TERMUACC
    UNIVERSAL
    NO SUBGROUPS
    NO USERS
```

Figure 13-2 Command to list the group GENUSERS

The group profile of a universal group contains limited group membership information. Only users who have group authorities other than USE, and those who have the group-SPECIAL, group-OPERATIONS or group-AUDITOR attribute, are added to the member list and will be listed in LISTGRP output, as shown in Figure 13-3. Therefore, the output of the LISTGRP command will not provide a complete list of group members and we can see that the other users don’t appear. The best way to list the members of a universal group is using the output of the database unload utility. See *z/OS V1R2.0 SecureWay Security Server RACF Security Administrator’s Guide*, SA22-7683 for further information.

```
CONNECT   MARTINS GROUP(GENUSERS) AUTHORITY(JOIN) UACC(ALTER)
CONNECT   GENADMI1 GROUP(GENUSERS) AUTHORITY(USE) UACC(ALTER)
CONNECT   GENADMI2 GROUP(GENUSERS) AUTHORITY(USE) UACC(ALTER)
LISTGRP   GENUSERS

------ output LISTGRP ------
INFORMATION FOR GROUP GENUSERS
    SUPERIOR GROUP=SYS1         OWNER=GENADMIN
    NO INSTALLATION DATA
    NO MODEL DATA SET
    TERMUACC
    UNIVERSAL
    NO SUBGROUPS
    USER(S)=      ACCESS=      ACCESS COUNT=      UNIVERSAL ACCESS=
      MARTINS       JOIN          000000               ALTER
         CONNECT ATTRIBUTES=NONE
         REVOKE DATE=NONE                 RESUME DATE=NONE
```

Figure 13-3 Commands connect and listgrp and output listgrp

The output from the database unload utility can be:

- Loaded into DB2, and sample DB2 queries can be used
  - Existing DB2 sample query to list all users in a group will list all the users in a UNIVERSAL group (see samplib member RACDBUQR).
  - New DB2 sample query will list all the UNIVERSAL groups and the number of members in each group.
- UserID as input to RACFICE
  - RACFICE tool (IRRICE in samplib) has a new sample query added to list all the members in a group (ID is GRPM).
  - New RACFICE query is also added, to list all the UNIVERSAL groups (ID is UGRP).

The **DELGROUP** command can not be used to delete a UNIVERSAL group. You should use the remove ID utility (IRRRID00) to delete the group and remove all member connections. If not, you will receive the following message:

```
ICH05008I WARNING: GENUSERS is a universal group.
```

Run the Remove ID utility to remove all users from the group. See *z/OS V1R2.0 SecureWay Security Server RACF Security Administrator's Guide*, SA22-7683 for information about using IRRRID00 to delete universal groups.

13.2.1 External changes

Following are changes related to group size expansion:

- R_admin callable service (IRRSEQ00)

  Keyword support is added for UNIVERSAL keyword on ADMN_ADD_GROUP function.
- SMF Records

  Type 6 relocate section for ADDGROUP updated to indicate UNIVERSAL group.
- Database Unload (IRRDBU00) Records

  Type 0100 record for GROUP base segment updated with new field for UNIVERSAL group.
- RACF Database Templates

  Flag field UNVLFG added to base segment of the group template.

13.2.2 Compatibility and migration

Down-level system sharing of the RACF database means that one system has support for UNIVERSAL groups, and one is down-level and does not accept the UNIVERSAL keyword or create UNIVERSAL groups.

- On down-level systems sharing the RACF database and executing commands, the commands execute as follows:
  - ADDGROUP with UNIVERSAL keyword will fail.
  - DELUSER and REMOVE will work as before.
  - DELGROUP will not issue any warning message.
  - DELGROUP will succeed if UNIVERSAL groups members all have authority higher than USE and/or have group-related user attributes.

  - ADDUSER and CONNECT will add users to the UNIVERSAL group, but they will be included in the 5957 limit.
  - ALTUSER, with AUTHORITY(USE) specified will include the member in the 5957 limit.
- An up-level system can be used to get an AUTH(USE) user not counted toward the 5957 limit.
  - If a user is connected with AUTH(USE) and has no group-related user attributes, and is showing up in LISTGRP output, issuing the CONNECT command with AUTH(USE) will change the member to not be counted towards the group member size limit. The following message is issued if the system has UNIVERSAL support but templates are down-level:

```
ICH577RE WARNING: BASE SEGMENT OF GROUP TEMPLATE AT LEVEL XXXXXX DOES NOT CONTAIN
FIELD UNVFLG
```

13.3 SAFTRACE tool

The SAFTRACE tool in z/OS V1 R2 provides the ability to trace all RACROUTEs, RACF callable services, and RACF Database Manager requests that go through the RACF routers. When tracing these services, the trace routine copies the parameter list into a GTF record before and after the service runs.

13.3.1 Using SAFTRACE

Examples of situations where SAFTRACE can be helpful are as follows:

- RACF database contention has been observed:
  - Trace DATABASE(ALTER) requests on the specific ASID indicated via GRS contention displays. Alter requests generally prevent readers (majority) from getting service.
- Excessive database I/O for a given address space:
  - Trace reads to see what CLASS/ENTITIES are related.
- Excessive verifies:
  - Set a trace on RACROUTE(TYPE(2,5,9)) and determine who is issuing all of the RACROUTE calls.

Trace points

What is being traced RACF, either through the SAF routers to the database or directly using the ICHEINTY interface, is the following:

- Through SAF routers ICHSFR00 and IRRSFR11:
  - Internal calls to the security product may not be traced. Internal calls are calls RACF makes to itself, such as RACINIT calling RACHECK.
  - All calls made via RACROUTE or Callable Service interfaces will be traced. Calls that issue SVC (pre-RACROUTE) or directly enter the security product will not be traced.
- RACF Database manager ICHEINTY interface:
  - All ICHEINTYs and internal security product calls to the database manager are traced.

13.3.2 Using the SET TRACE command

The **set trace** command specifies whether or not Generalized Trace Facility (GTF) records should be created for the specified events. If the TRACE operand is specified, at least one suboperand is required. The related record is EF44 for each trace event. See *z/OS MVS Interactive Problem Control System (IPCS) User's Guide*, ST03-0473 or *z/OS MVS Diagnosis: Tools and Service Aids*, GA22-7589 for information on viewing these records. The **set trace** command syntax is as follows:

```
SET TRACE(APPC | NOAPPC | ASID | NOASID | CALLABLE | NOCALLABLE | DATABASE | NODATABASE
| IMAGE | NOIMAGE | JOBNAME | NOJOBNAME | RACROUTE | NORACROUTE)
```

Note: These trace records are intended only for diagnostic use when requested by the IBM support center. The format will not be documented.

Command tracing

Command tracing can be useful when diagnosing command errors, and can be used with any command in *z/OS SecureWay Security Server RACF Command Language Reference*, SA22-7687 that supports the AT keyword. It provides a step-by-step history of how the command text is parsed and rebuilt by the RACF command envelope module and can be used to determine which TSO macros are used. It can help determine if the problem is in the command envelope module or the command processor load module, and also helps determine where a failure occurred during command parsing and rebuilding.

To obtain a command trace, do the following:

1. At the operator console, activate **SET TRACE** (suboperand), including all appropriate additional suboperands, as follows:

   ```
   SET TRACE (RACROUTE(TYPE(3 5 9)))
   ```

2. From a TSO terminal in TSO READY mode, or from ISPF Option 6 (Command), do one or more of the following:

   a. Issue a RACF command and append the two characters **-c** (note there is a leading blank between the command and the two characters) to obtain a trace of the command buffer. This keyword is the most useful and is recommended over the others.

   b. Issue a RACF command and append the two characters **-t** (note there is a leading blank between the command and the two characters) to obtain a trace of the TSO macros used during the parsing of the RACF command and to obtain a trace of the command buffer.

   c. Issue a RACF command and append the two characters **-n** (note there is a leading blank between the command and the two characters) to not run the command.

3. Output is sent to the user’s TSO terminal.

See *z/OS SecureWay Server RACF Diagnosis Guide*, GA22-7689 for more details.

Attention: Trace records might contain passwords; therefore, trace output data sets should be appropriately protected.

13.3.3 RACF tracing

RACF provides a trace facility that allows tracing of RACROUTEs, Callable Services, and RACF Database Manager requests. When tracing these services, the trace routine copies the parameter lists into a GTF record before and after the function executes. IPCS is used to view the trace data.

GTF

The generalized trace facility (GTF) is a service aid you can use to record and diagnose system and program problems. GTF is part of the MVS system product, and you must explicitly activate it by entering a **`START GTF`** command.

Using GTF, you can record a variety of system and program events on all of the processors in your installation. However, because GTF uses more resources and processor time than system trace, IBM recommends that you use GTF when you experience a problem, selecting one or two events that you think might point to the source of your problem. This will give you detailed information that can help you diagnose the problem. You can trace combinations of events, specific incidences of one type of event, or user-defined program events that the GTRACE macro generates. For more details see *z/OS MVS Diagnosis: Tools and Service Aids*, GA22-7589 and *z/OS SecureWay Server RACF Diagnosis Guide*, GA22-7689.

GTF and IPCS

You can use IPCS to merge, format, display, and print GTF output of the TRACE. See *z/OS MVS Interactive Problem Control System (IPCS) User's Guide*, ST03-0473 and *z/OS MVS Interactive Problem Control System (IPCS) Commands*, SA22-7594 for information about the COPYTRC, GTFTRACE, and MERGE subcommands, and the trace processing option of the IPCS dialog.

To obtain the RACF traces of RACROUTEs, Callable Services, and RACF Database Manager requests, do the following:

1. Start the GTF using the GTFRACF, as shown in Figure 13-4 and Figure 13-5:

   ```
   START GTFRACF.GTF,,,NOPROMPT
   ```

2. Use the SET command to enable your trace:

   ```
   @SET TRACE(RACROUTE(TYPE(5)) JOBNAME(IBMUSER))
   ```

3. Reproduce the scenario that you desire,. For example: start batch job, login, start application, use CICS application or access resource.
4. Next, stop GTF to prevent excessive traces.

   ```
   STOP GTF
   ```

5. Use IPCS to view the trace data. The input trace data is contained in the data set specified on the IEFRDER DD card in the GTFRACF (or other) procedure. The sample GTFRACF procedure specifies `SYS1.TRACE`. Once the TSO IPCS session is active the IPCS subcommand `IP GTF USR` may be used to display the formatted trace.

```
//GTFRACF PROC MEMBER=GTFPRM#O
//BR14 EXEC PGM=IEFBR14,REGION=512K
//SYSPRINT DD SYSOUT=*
//D DD DISP=(OLD,DELETE),UNIT=3380,VOL=SER=TEMP01,
// DSN=SYS1.TRACE
//IEFPROC EXEC PGM=AHLGTF,PARM=©MODE=EXT,DEBUG=NO,SA=100K,AB=100K©,
// REGION=2880K,TIME=NOLIMIT
//IEFRDER DD DSNAME=SYS1.TRACE,UNIT=3380,VOL=SER=TEMP01,
// DISP=(NEW,CATLG),SPACE=(TRK,(100))
//SYSLIB DD DSNAME=RACFDRVR.PARMLIB.R6(&MEMBER),DISP=SHR
```

Figure 13-4 Sample proclib member for GTF

```
TRACE=USRP
USR=(F44),END
```

Figure 13-5 Sample parmlib member: GTFPRM#O

13.3.4 Trace performance considerations

Security as implemented on the z/OS platform includes many calls to the security product. This trace facility can adversely affect system performance by adding to the path length associated with performance-sensitive security functions. This trace should only be used as a debugging aid. Caution should be exercised when designing the trace (as with any other trace) so as to impose the least performance penalty. For example, if the address space ID or jobname is known, use these to restrict the scope of the trace.

Usage considerations

When using RACF tracing, consider the following:

- The RACF subsystem must be up and running.
- GTF must be active.
- Trace information is not saved across IPLs.
- For OMVS calls, you need an asterisk (*) in the jobname filter to trace spawned processes. Otherwise, you will not get a complete set of records.
- RACF database contention:
  - The scope of RACF database serialization is dependent on the RACF database manager parameter list. Set up a trace for Manager ALTER requests on the specific ASID indicated via GRS contention displays. Examine the trace records to find out what RACF is changing in the database so frequently.
  - When applications specify subsystem and requestor information, this will be contained in the trace record header. You can determine who is issuing the security function.

Reading a trace output

The trace output formatted by IPCS is split into three main sections. The first section contains common information for all services, for example, the caller's information and return codes. In the second section the parameter lists and the unloaded parameters are found. Each of these two sections starts with a length field and is followed by the data itself. The third section is a complete hex dump of the entire GTF record which includes the header information, parameter lists and all of the parameters that were unloaded.

For RACROUTE requests there are two parameter lists. The first one is the SAF parameter list mapped by ICHSAFP. The second parameter list is mapped by the specific RACROUTE type parameter list. These are described in *z/OS SecureWay Security Server RACF Data Areas*, GA22-7680. For Callable Services, the parameter lists can be found in *z/OS SecureWay Security Server RACF Callable Services*, SA22-7687. For Manager calls, see the *z/OS SecureWay Server RACF Diagnosis Guide*, GA22-7689 for the Manager parameter list mapping.

Following the parameter lists are the parameters that are unloaded. Not all parameters are unloaded. Before most parameters are unloaded there are extra entries that say OFFSET##. This is for informational purposes, to let you know that the following parameter is at offset ## in the parameter list. This helps determine what parameter you are examining.

Due to nesting of some services PRE and POST trace records might not be in sequential order. For example, one might see two PRE calls and then two POST calls.

13.4 Support of mixed case profile names

The profiles in the RACF database contain the information RACF needs to control access to resources. The RACF commands allow you to add, change, delete, and list the profiles for:

- Users
- Groups
- Data sets

General resources, which include terminals, DASD volumes, and all other resource classes are defined in the RACF class descriptor table (CDT). RACF commands allow you to list, modify, add, and delete profiles for users, groups, connect entries, and resources.

Mixed case profile names are necessary in order to allow Enterprise Java Bean (EJB) authorization roles (EJB roles must allow mixed case), in RACF profiles for:

- WebSphere and CICS
- EJB compliance test suite

Mixed case profile names are now accepted and preserved. So, in this way, the applications can now allow use of mixed case profile names in RACF classes. In this section we discuss this support with respect to the following topics:

- RACF macro changes
- New RACF classes
- Enhanced commands
- Enhanced panels
- Migration considerations

13.4.1 RACF macro changes

RACF customization macros provide information on the macros that are part of the RACF product and are necessary to its operation. System programmers can use these macros to tailor RACF to meet the needs of your installation in various ways. For example, they can add additional classes to the class descriptor table via the ICHERCDE macro.The ICHERCDE macro generates entries for the resource class descriptor table (CDT). The class descriptor table contains information that directs the processing of general resources. The table consists of an entry for each class except USER, GROUP, and DATASET.

In order to support mixed case profile names, the new keyword CASE=UPPER | ASIS was added to ICHERCDE macro. It specifies whether mixed-case profile names are allowed for the class specified by the CLASS operand. UPPER is the default. When ASIS is specified, RACF commands preserve the case of profile names for the specified class. Lowercase characters are allowed in any position of the profile name where alphabetic characters are allowed, based on the character restrictions specified in the FIRST= and OTHER= operands.

13.4.2 New RACF classes

Two new RACF classes are added for EJBs, to enable use of mixed case profile names, as follows:

EJBROLE	Member class for Enterprise Java Beans authorization roles.
GEJBROLE	Grouping class for Enterprise Java Beans authorization roles.

13.4.3 Enhanced commands

The following changes have been made in order to enable mixed case profile names. For more information see *z/OS V1R2.0 SecureWay Security Server RACF Command Language Reference*, SA22-7687.

The **RDEFINE, RALTER, RLIST, RDELETE** and **PERMIT** commands have been enhanced to accept mixed case profile names for classes whose CDT entries are specified with CASE=ASIS.

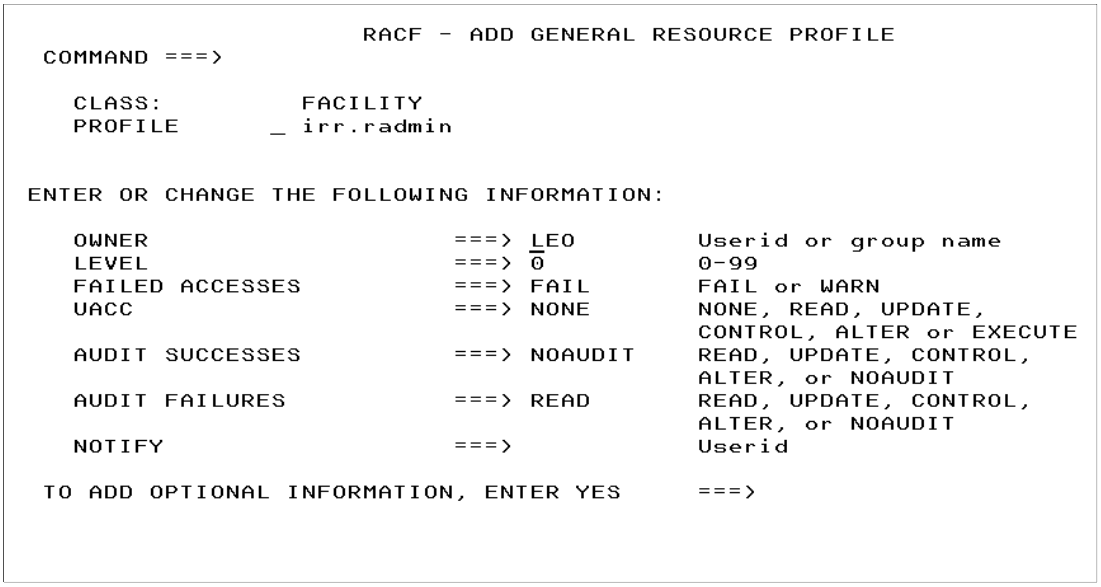
The **RDEFINE** and **RALTER** commands have been enhanced to accept mixed case member names in the **ADDMEM** and **DELMEM** operands for mixed case classes.

The **ADDSD, RDEFINE** and **PERMIT** commands have been enhanced to accept mixed case profile names in the **FROM** operand when the **FCLASS** operand specifies a mixed case class.

The **SEARCH** command has been enhanced to support mixed case strings on the **CLIST, FILTER** and **MASK** operands for mixed case classes.

13.4.4 Enhanced panels

The only external change that you will see is the profile name in the case in which it was entered, regardless of the class, as you navigate through the panels. This is illustrated in Figure 13-6.

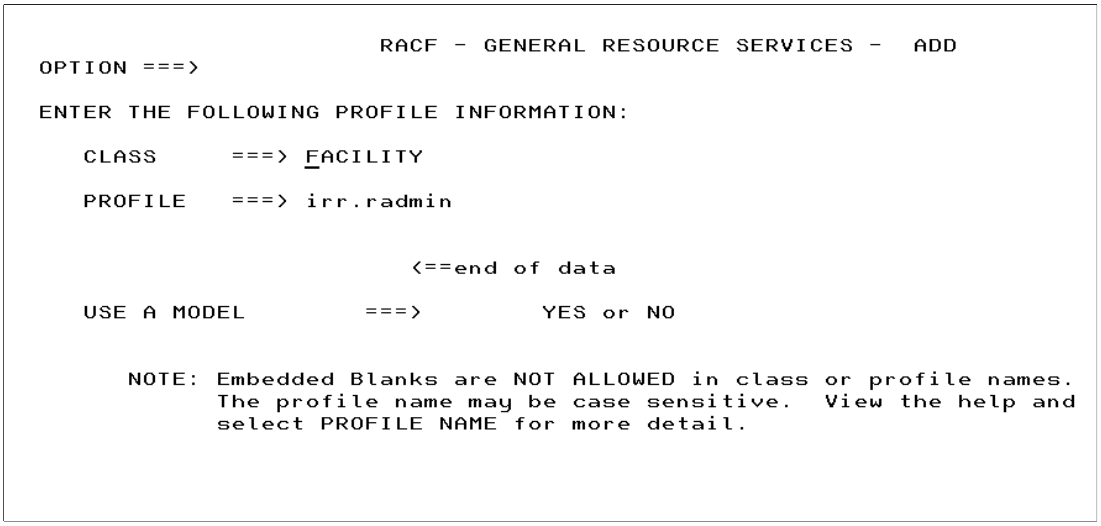


```
                       RACF - ADD GENERAL RESOURCE PROFILE
  COMMAND ===>

    CLASS:         FACILITY
    PROFILE      _ irr.radmin


 ENTER OR CHANGE THE FOLLOWING INFORMATION:

    OWNER                    ===> LEO        Userid or group name
    LEVEL                    ===> 0          0-99
    FAILED ACCESSES          ===> FAIL       FAIL or WARN
    UACC                     ===> NONE       NONE, READ, UPDATE,
                                             CONTROL, ALTER or EXECUTE
    AUDIT SUCCESSES          ===> NOAUDIT    READ, UPDATE, CONTROL,
                                             ALTER, or NOAUDIT
    AUDIT FAILURES           ===> READ       READ, UPDATE, CONTROL,
                                             ALTER, or NOAUDIT
    NOTIFY                   ===>            Userid

  TO ADD OPTIONAL INFORMATION, ENTER YES     ===>
```

Figure 13-6 Profile name in lower case

The RACF command that is eventually issued makes sure that the command text is in upper case.

Help panels have been modified and added to explain this behavior, as you can see in the NOTE in Figure 13-7.



```
                         RACF - GENERAL RESOURCE SERVICES -  ADD
OPTION ===>

ENTER THE FOLLOWING PROFILE INFORMATION:

   CLASS     ===> FACILITY

   PROFILE   ===> irr.radmin


                           <==end of data

   USE A MODEL         ===>        YES or NO


      NOTE: Embedded Blanks are NOT ALLOWED in class or profile names.
            The profile name may be case sensitive.  View the help and
            select PROFILE NAME for more detail.
```

Figure 13-7 NOTE: New help panels

13.4.5 Migration considerations

The following items must be carefully taken into account.

- No existing RACF classes were changed to CASE=ASIS, and external changes to RACF classes are not supported.
- Users can use CASE=ASIS for their own new or existing classes.
  - Users must keep CDTs in synch across nodes when using RACF Remote sharing Facility (RRSF).
  - When changing an existing class to CASE=ASIS, educate the users, because RACF will accept profile names in the case in which they are typed.

> **Note:** If users are used to typing in lower case and relying on RACF to upper-case profile names, then RACF will be creating profiles which they do not expect.

13.5 Support for DB2 version 7

There is new support introduced in RACF to support DB2 Version 7 features that affects the RACF-supplied DB2 external security module, as follows:

- New RACF support for DB2
- DB2 Version 7 new features
- New and changed RACF messages
- Installation concerns

13.5.1 New RACF support for DB2

To protect data and resources associated with a database server, DB2 uses a combination of external security services and internal access control information. DB2 calls the RACF DB2 external security module during:

- DB2 initialization, to create in-storage profiles
- DB2 authorization, to check DB2 objects and authorities
- DB2 termination, to delete in-storage profiles

To support those DB2 new features, the RACF-supplied sample DB2 external security module **(‘SYS1.SAMPLIB(IRR@XACS)’)** was changed. The changes were:

- Two new RACF general resources classes:
  - **MDSNJR**
  - **GDSNJR**
- Code to process the specification of a database name on a **ALTER INDEX** and **DROP INDEX** request if the **XAPLCRVW** bit is on.
- Code to process the specification of a set of database names on a **CREATE VIEW** request if the **XAPLCRVW** bit is on.
- A new exit option **(&ERROROPT)** is introduced to support the new DB2 reason code.

> **Attention:** There is no support for DBNAME on CREATE ALIAS.

- **&ERROROPT** instructs DB2 what to do when an unexpected error occurs in the DSNX@WAC exit. An unexpected error is:
  - An abend occurs in DSNX@XAC
  - An unexpected return code is return by DNSX@XAC
  - DNSX@XAC instructs DB2 to not call it again:
    - &ERROROPT=1

      Defer to DB2 when an unexpected error occurs. This is the default.
    - &ERROROPT=2

      The DB2 subsystem is instructed to terminate if an unexpected error occurs. This is only effective if DB2 Version 7 is the invoker.

For more details about these changes and others, see *z/OS V1R2 SecureWay Security Server RACF Administration Guide*, SA22-7683.

13.5.2 DB2 Version 7 new features

Version 7 of DB2 UDB Server for z/OS offers several features that help you integrate, analyze, summarize, and share data across your enterprise. The new features are as follows:

- The **JAR** object (JAVA archives).
- The **USAGEAUTJ** privilege name.
  - The XAPLPRIV privilege name is USAGEAUTJ

When a user owns the Java archive (JAR), then XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If the user does not own the JAR, the user must have sufficient authority to one of the following RACF profiles:

RACF class	Profile name
MDSNJR	DB2-subsystem.schema-name.JAR-name.USAGE

GDSNJR	DB2-subsystem.schema-name.JAR-name.USAGE
DSNADM	DB2-subsystem.SYSADM

13.5.3 DBADM privilege

Database names passed on DBADM privilege can drop and alter any table space, table, or index in the database and issue a COMMENT ON, LABEL ON, or LOCK TABLE statement for any table. If the value of field DBADM CREATE VIEW on installation panel DSNTIPP was set to YES during DB2 installation, a user with DBADM authority can:

- Create a view for another user ID. The view must be based on at least one table and that table must be in the database where the user ID that issued the CREATE VIEW statement has DBADM authority.
- Create an alias for another user ID on any table in the database.

However, a user with DBADM authority on one database can create a view on tables and views in that database and other databases if the authorization ID for which the view is created has all other privileges that are required to create the view.

A user with DBADM authority can grant those privileges to others, but cannot create a view on a view that is owned by another user ID.

- DB2 DSNX@XAC supports a reason code of x'10' (16) on initialization. This new reason code instructs the DB2 subsystem to terminate if:
  - An abend occurs in the DSNX@XAC exit
  - The DSNX@XAC exit instructs DB2 to no longer call it
  - An unexpected return code is returned to DB2 from DSNX@XAC exit

For more details about these changes and others, see *DB2 Universal Database for OS/390 and z/OS Administration Guide Version 7*, SC26-9931.

13.5.4 New and changed RACF messages

Two new messages have been added and five messages have been changed. For more details about RACF messages, see *z/OS SecureWay Security Server RACF Messages and Codes*, SA22-7686.

New RACF messages

The following messages are added:

```
IRR912I NATIVE DB2 AUTHORIZATION IS USED
IRR913I DB2 SUBSYSTEM TERMINATION REQUESTED
```

Changed RACF messages

The following messages are changed:

```
IRR900A RACF/DB2 EXTERNAL SECURITY MODULE FAILED TO INITIALIZE FOR DB2 SUBSYSTEM
subsystem-name BECAUSE CLASS classname COULD NOT BE RACLISTED. RACROUTE RETURN CODE
return_code, RACF RETURN CODE return_code, REASON CODE reason_code.

IRR901A RACF/DB2 EXTERNAL SECURITY MODULE FAILED TO INITIALIZE FOR DB2 SUBSYSTEM
subsystem-name BECAUSE NO ACTIVE DB2 RELATED CLASSES WERE FOUND.

IRR902A RACF/DB2 EXTERNAL SECURITY MODULE FAILED TO INITIALIZE FOR DB2 SUBSYSTEM
subsystem-name BECAUSE THE INPUT ACEE WAS {MISSING|/NOT VALID}
```

```
IRR903A RACF/DB2 EXTERNAL SECURITYMODULE FAILED TO INITIALIZE FOR DB2 SUBSYSTEM
subsystem-name BECAUSE RACF WAS NOT ACTIVE.

IRR909I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM subsystem-name IS USING
OPTIONS: &CLASSOPT= classopt &CLASSNMT= classnmt &CHAROPT= charopt &ERROROPT= erroropt
&PCELLCT= pcellct &SCELLCT= scellct
```

13.5.5 Installation concerns

There are no changes to the current **IRR@XACS** installation process; however, be aware that:

- DBADM will be used if the DB2 installation parameter DBADM CREATE ATH is set to YES.
- Default &ERROROPT=1 ensures that it is business as usual for those who make no changes to **IRR@XACS**.

13.6 Kerberos extensions

The z/OS SecureWay Network authentication server requires a registry of information about principals and realms. This security information is stored in RACF User and General Resource profiles. The KERB segment of the user profile is used to store information about Network Authentication Service principals on your local system. The general resource class KERBLINK allows you to map principals to RACF user IDs on your system. The general resource class REALM defines the local Network Authentication Service realm and its trust relationships with foreign realms. Kerberos administration is done via RACF commands or panels. For more information about this, see *z/OS SecureWay Security Server RACF Security Administrator's Guide*, SA22-7683. In this section, we discuss the following topics:

- Kerberos registry support overview
- Commands keyword updates
- Other updates
- Dependencies and migration considerations
- Benefits

13.6.1 Kerberos registry support overview

Network Authentication Service uses RACF to store and administer information about principals and realms. RACF commands/panels are used for Kerberos administration. REALM class profiles are used to define information about the local Kerberos realm and foreign realms. You must define your local realm to RACF before you define local principals. This is because the local realm name is used to generate keys for local principals. You define your local realm by creating a profile in the REALM class called KERBDFLT. Using the KERB option of the RDEFINE and RALTER commands, you can specify the following information about your local realm:

KERBNAME	Minimum ticket lifetime for the local realm
DEFTKTLFE	Default ticket lifetime for the local realm
MAXTKTLFE	Maximum ticket lifetime for the local realm
ENCRYPT	Key encryption options (DES, DES3, DESD)
PASSWORD	Value of the password for the local realm

Note: This password is not a RACF user password. Therefore, it is not constrained by SETROPTS password rules that may be specified to control user passwords. In addition, the installation-defined new-password exit (ICHPWX01) is not invoked.

A password value must be supplied.

Uppercase and lowercase letters are accepted and maintained in the case in which they are entered.

RACF KERB segment

Local Kerberos principals are defined as RACF users with a KERB segment. The KERB segment contains the information about a Network Authentication Service user, such as the local principal name that will be mapped to this RACF user ID. In order to define local principals as RACF users, use the ADDUSER and ALTUSER commands with the KERB option. This creates a KERB segment in the user profile. You can specify the following information in this segment:

KERBNAME	User's local principal name
MAXTKTLFE	User's maximum ticket life
ENCRYPT	User's key encryption types (DES, DES3, DESD)

Important: The KERB segment of the RACF user profile defines a user as a local principal. If KERB segment information is directed to a remote RRSF node, users will be defined as local principals on all Network Authentication Service servers that share that RACF database.

Using RACF with SecureWay Network Authentication Service on z/OS Version 1 Release 2, allows more encryption types for the following keys:

- DES
- DES3 (Triple DES)
- DESD (DES with Derivation)

Allow or disallow each type on a per-profile basis, enabled via `AU/ALU` or `RDEF/RALT`.

13.6.2 Command keyword updates

The following RACF commands are changed to support the new KERB keyword:

- ADDUSER
- ALTUSER
- RALTER
- RDEFINE

The KERB keyword has the following format:

```
KERB(ENCRYPT(option) | NOENCRYPT)
```

The ENCRYPT keyword option or subkeywords are as follows:

- **`DES | NODES`**
- **`DES3 | NODES3`**
- **`DESD | NODESD`**

The **LISTUSER** and **RLIST** commands display new information for the **KERB** attribute:

```
KEY ENCRYPTION TYPE=DES DES3 DESD
```

The **SETROPTS** command supports the following new keyword:

```
KERBLVL(0|1)
```

This keyword specifies what level of encryption processing should occur when a KERB segment is being processed for user and general resource profiles.

0 Specifies that only DES keys should be created and the ENCTYPE fields will be set to X'00000001'. The value of the ENCRYPT field will be ignored by R_kerbinfo.

1 Specifies that keys should be created for DES, DES3 and DESD. This level also allows for the enablement and disablement of these key types for a specific profile through the use of the ENCRYPT field.

> **Note:** If the value of KERBLVL is reset from a value of 1 to 0, some processing may not perform as expected. When this is done the listing of profiles that had DES3 or DESD encryption enabled, or DES encryption disabled, will still show this as is denoted by the profile settings. The IBM Kerberos server (through the use of the R_kerbinfo call) will process as if DES is enabled and the others are disabled, regardless of whether the profile is defined to have these encryption types enabled or disabled. Setting KERBLVL to a value lower than the current value should be avoided due to this inconsistency.

13.6.3 Other updates

The z/OS Version 1 Release 2, introduces other updates to RACF Kerberos extensions, as follows:

- The value fields CURKEY and PREVKEY returned for information retrieval function codes X’01’ and X’04’ have a new format. Callable service, R_kerbinfo (IRRSMK00), is changed to support the new format for value fields CURKEY and PREVKEY.
- Callable service, R_admin (IRRSEQ00), is changed to support a new field name. ENCRYPT, for the user, resource, and system options functions KERB segment fields table.
- The database templates were updated. The ENCRYPT field was added to the KERB segment of the following templates in support of SETROPTS KERBLVL processing for Network Authentication Service:
  - USER
  - GENERAL
- The dynamic-parse specification data (IRRDPSDS) was updated. The **ENCRYPT** keyword was added, with the following valid settings:
  - **DES | NODES**
  - **DES3 | NODES3**
  - **DESD | NODESD**
- The RACF SMF data unload utility unload new **SETROPTS KERBLVL** value.

For more information see *z/OS SecureWay Security Server RACF Security Administrator’s Guide*, SA22-7683 and *z/OS SecureWay Security Server RACF Security System Programmer’s Guide*, SA22-7681.

13.6.4 Dependencies and migration considerations

The Network Authentication Service V1R2 server must be installed before defining any keys. Ensure that all systems sharing your RACF database support the SETROPTS KERBLVL command before enabling KERBLVL(1). KERBLVL(0) is the default value. As an alternative, you may enable KERBLVL(1) if you ensure that no keys are generated or used from downlevel. Do not upgrade to level 1 until all systems sharing the database have multiple key code levels.

13.6.5 Benefits

The RACF Kerberos extensions provide the following benefits:

- Triple DES and DES with derivation in addition to DES encryption
- Flexibility in choice of encryption algorithms
- Ability to restrict use of certain encryption types
- Encryption controlled on a per-profile basis
- Extends existing administration interfaces
- Enables you to set up encryption policy before cutting over to a new support level

14

Firewall technology enhancements

This chapter introduces the new enhancements of the Firewall Technologies contained within z/OS Version 1 Release 2. The following items will be discussed:

- Configuration server enhancements
- DNS configuration support - REMOVAL
- ISAKMP server enhancements
- Migration considerations

14.1 Firewall technology

A *firewall* is a network component that allows a single point of access between a private network and a public network. There are two main types of firewalls: packet filtering and proxy.

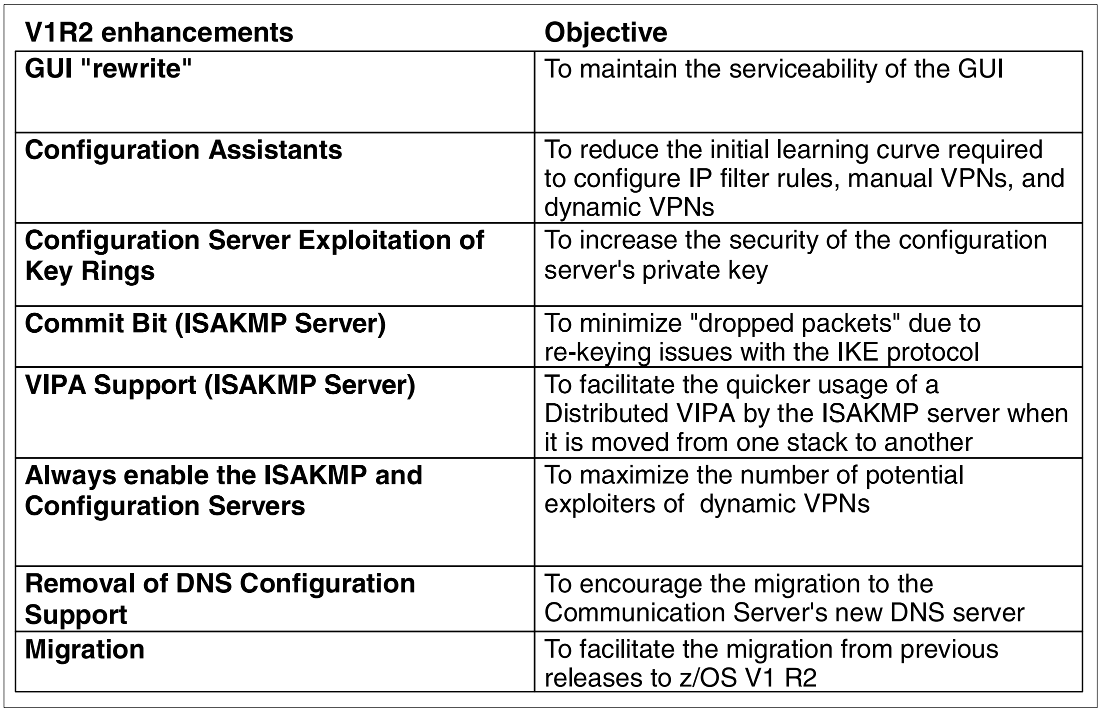
- A *packet filtering* firewall examines all traffic routed between the two networks to see if it meets certain criteria. If it does, it is routed between the two networks; if it does not, it is stopped.
- A *proxy* firewall filters specific types of network traffic. This is also known as protocol filtering because the decision to forward or reject traffic depends on the protocol that is used.

In order to control access between the private network (intranet) and the public network (usually the Internet), and at the same time permit authorized transactions, z/OS Firewall Technologies provides three main technologies: network servers, filters and address translation, and virtual private network. z/OS Firewall Technologies are tools you use to implement different firewall architectures. For more details see *z/OS SecureWay Security Server Firewall Technologies*, SC24-5922.

Once you choose your security policy and your firewall architecture, you can select the necessary firewall tools to implement them. The Firewall Technologies contained within z/OS Version 1 Release 2 have been enhanced. The requirement of possessing a Security Server License in order to use the ISAKMP and Configuration servers was removed. Now, non-security servers can use the Configuration server (and GUI) when configuring IP filtering, NAT, VPNs, and dynamic VPNs.

14.2 Configuration server enhancements

The configuration server is the configuration client’s graphical user interface (GUI) to the Firewall. This server processes requests from the AIX, Windows 95 or Windows NT configuration clients. Once it is set up, it is considered part of the Firewall machine. This topic identifies the enhancements made on the configuration server in order to provide a better way to perform configuration functions and improve it. In this topic, the following items are described as shown in Figure 14-1 on page 235:



V1R2 enhancements	Objective
GUI "rewrite"	To maintain the serviceability of the GUI
Configuration Assistants	To reduce the initial learning curve required to configure IP filter rules, manual VPNs, and dynamic VPNs
Configuration Server Exploitation of Key Rings	To increase the security of the configuration server's private key
Commit Bit (ISAKMP Server)	To minimize "dropped packets" due to re-keying issues with the IKE protocol
VIPA Support (ISAKMP Server)	To facilitate the quicker usage of a Distributed VIPA by the ISAKMP server when it is moved from one stack to another
Always enable the ISAKMP and Configuration Servers	To maximize the number of potential exploiters of dynamic VPNs
Removal of DNS Configuration Support	To encourage the migration to the Communication Server's new DNS server
Migration	To facilitate the migration from previous releases to z/OS V1 R2

Figure 14-1 Overview of z/OS V1R2 Firewall Technologies enhancements

14.2.1 GUI “rewrite”

The GUI “rewrite” was a partial rewrite. For the most part, only the presentation layer has been changed. The interactions between the GUI and the configuration server remain essentially the same, as shown in Figure 14-2 on page 236. The GUI is migrated from Java 1.1.8 runtime to Java 1.3, in order to fix threading issues and to restructure the presentation logic. Also, by moving to the IBM Unity toolkit, this eliminates any dependencies on the non-IBM toolkits, like Microline and JScape, as well as the use of newly deprecated functions.

The Unity toolkit should provide a more consistent look with other IBM products. It is supported on a number of platforms, which allows support of more platforms such as Win 2k, Win Millennium, and Linux (as supported by the Hursley JDK). In the process of moving to the Unity toolkit, the latest SSL support is used. The mechanism of invoking the GUI has not changed.

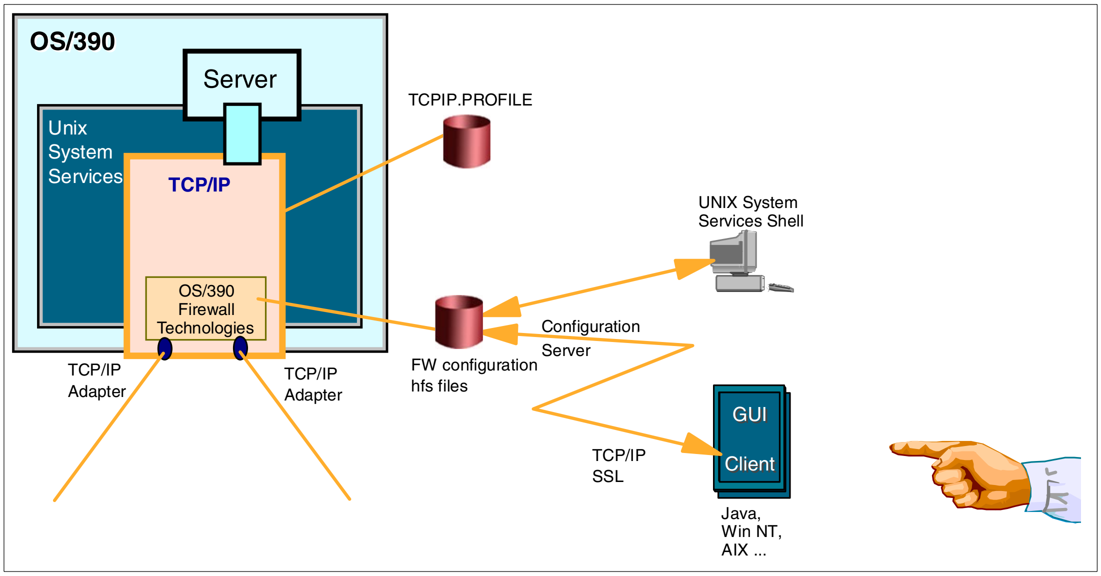
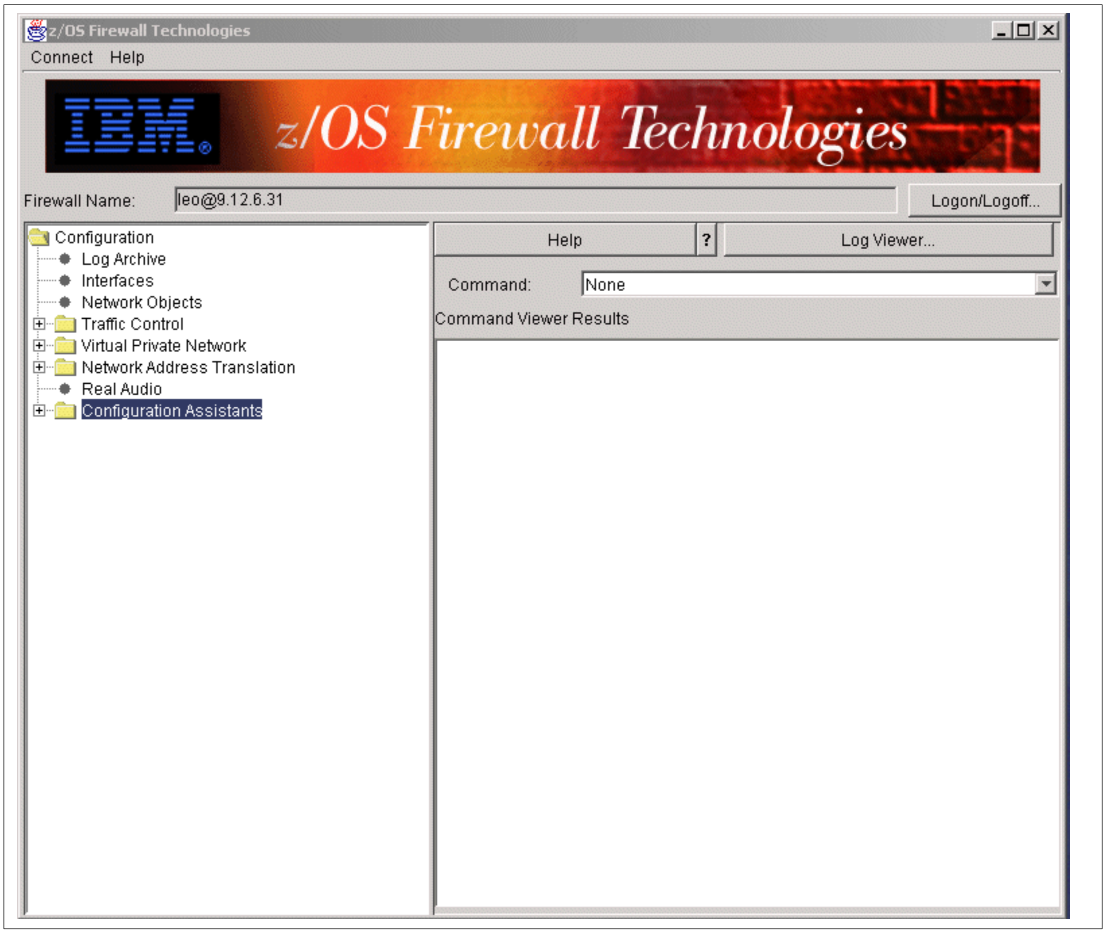


Figure 14-2 VPN configuration GUI rewrite

14.2.2 Configuration Assistants

Configuration Assistants (CAs) were introduced into the GUI, as you can see on Figure 14-3.



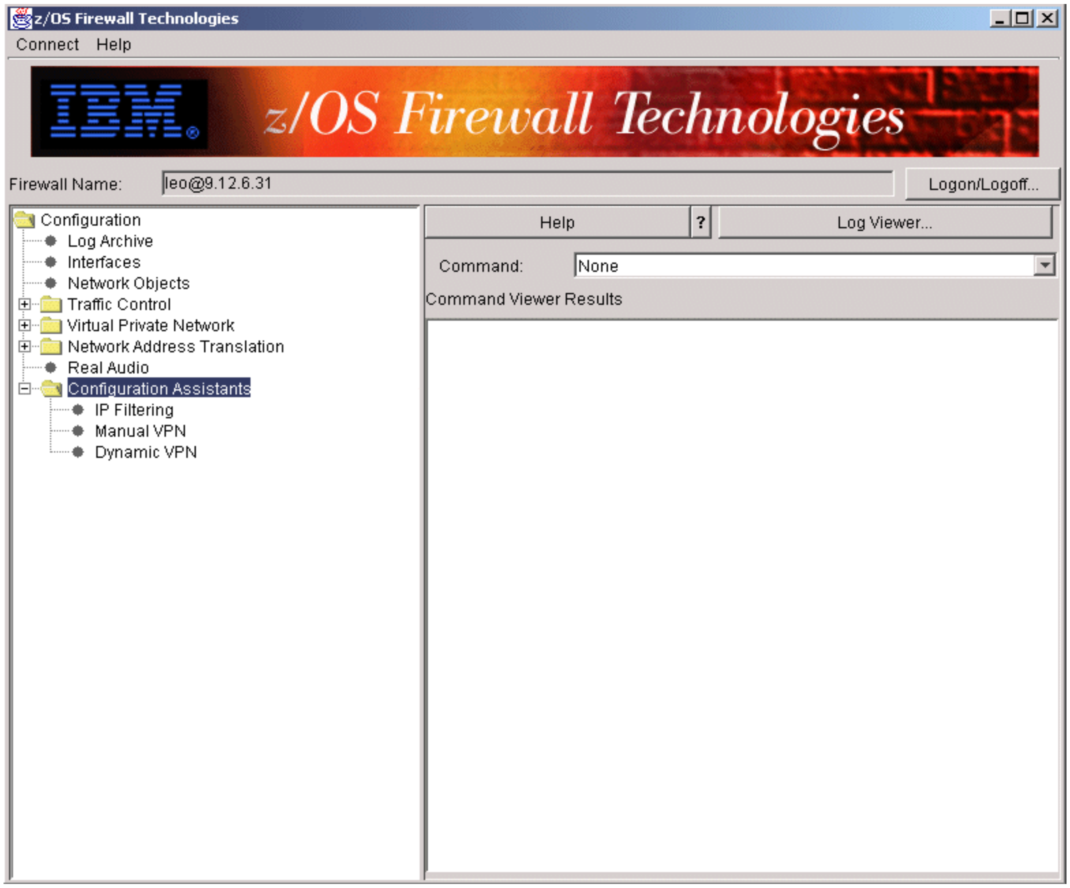
Firewall Name: leo@9.12.6.31

Command: None

Figure 14-3 Configuration Assistants in the GUI

Configuration Assistants are dialogs in the GUI that will guide one through the process of defining IP filters, manual VPNs, and dynamic VPNs; refer to Figure 14-4.

Configuration Assistants can be viewed as a combination of a wizard, help, and normal GUI panels. They are not intended to reduce the number of steps required to configure IP filters, manual VPNs and/or dynamic VPNs dynamic VPNs. They are intended to help one learn how to perform such configurations while actually performing such configurations.They also provide a faster path for an experienced user who wishes to create a series of new configuration definitions by allowing the ‘‘list” panels to be bypassed. The new structure of the GUI made it easy to provide the Configuration Assistant dialogs.



Firewall Name: leo@9.12.6.31

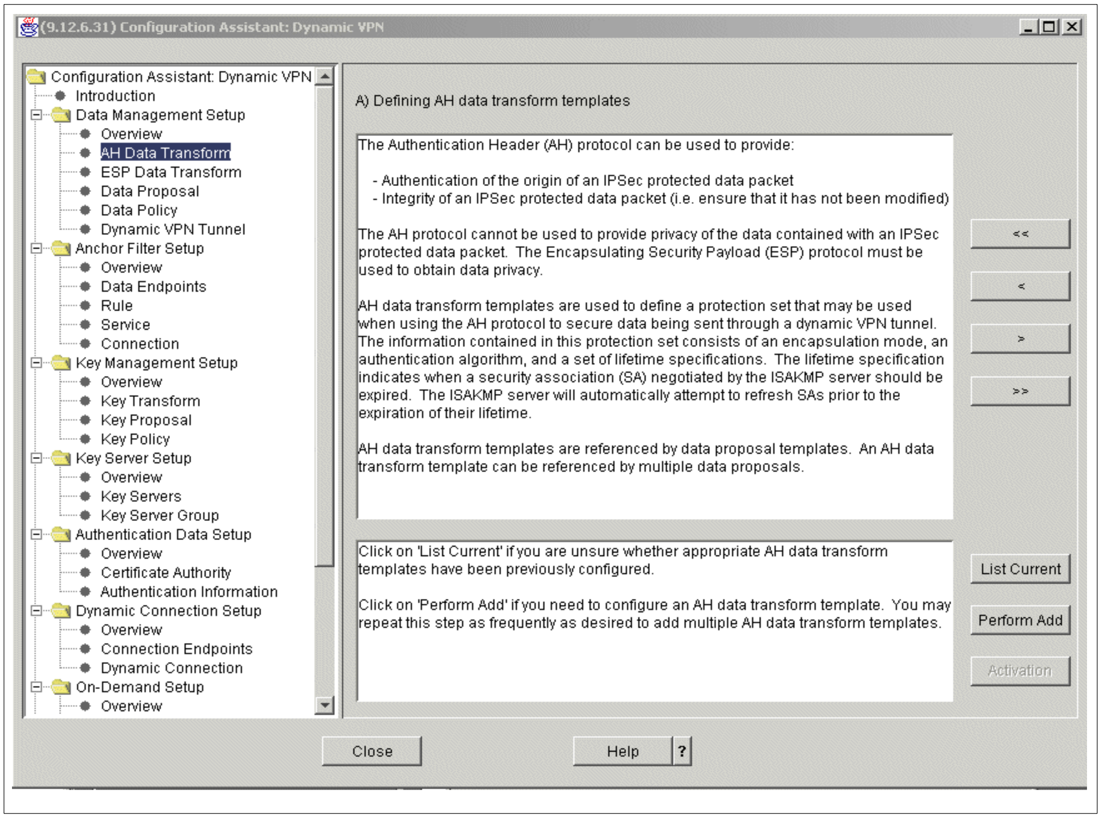
Command: None

Figure 14-4 Configuration Assistants - contents

Sample Configuration Assistant

A sample Configuration Assistants screen is shown in Figure 14-5 on page 238. The <<, >>, > and < buttons are used to navigate through the steps. The << and >> buttons get you to the previous step and the next step respectively, and the < and > buttons get you to the previous sub-step and the next sub-step, respectively. Also, steps and sub-steps can be directly accessed using the tree displayed on the left.

The upper text box alongside these buttons provides information about the step or sub-step being performed. The lower text box explains the usage of the buttons displayed at its right. These buttons are step- or sub-step specific; they typically provide a list function and an add function.



A) Defining AH data transform templates

The Authentication Header (AH) protocol can be used to provide:

- Authentication of the origin of an IPSec protected data packet
- Integrity of an IPSec protected data packet (i.e. ensure that it has not been modified)

The AH protocol cannot be used to provide privacy of the data contained with an IPSec protected data packet. The Encapsulating Security Payload (ESP) protocol must be used to obtain data privacy.

AH data transform templates are used to define a protection set that may be used when using the AH protocol to secure data being sent through a dynamic VPN tunnel. The information contained in this protection set consists of an encapsulation mode, an authentication algorithm, and a set of lifetime specifications. The lifetime specification indicates when a security association (SA) negotiated by the ISAKMP server should be expired. The ISAKMP server will automatically attempt to refresh SAs prior to the expiration of their lifetime.

AH data transform templates are referenced by data proposal templates. An AH data transform template can be referenced by multiple data proposals.

Click on 'List Current' if you are unsure whether appropriate AH data transform templates have been previously configured.

Click on 'Perform Add' if you need to configure an AH data transform template. You may repeat this step as frequently as desired to add multiple AH data transform templates.

Figure 14-5 Configuration Assistants dialog screen

In summary, Configuration Assistants simplify the configuration process in the sense that it helps the first time/occasional user by identifying configuration steps and displaying text that describes each step, and it helps the experienced user by providing fast path add panels.

14.2.3 Exploitation of key rings

Certificate authorities are trusted organizations that verify information about client users and then issue digital certificates that may be accepted by applications as authentication of client identities when used in a secure handshaking protocol such as Secure Sockets Layer (SSL). Trusting a certificate issued by a certificate authority is analogous to accepting a passport issued by a national passport agency as proof of identity; we trust that the agency has taken proper measures to verify the identity of the bearer of the passport.

In a similar manner, application servers may accept certificate-authority certificates. The firewall configuration server uses the Secure Sockets Layer (SSL) protocol of z/OS for communication between the graphical user interface (GUI) clients and the server.

The firewall configuration server must be configured to use the SSL option. Because the clients can only connect to the server using SSL, failure to configure the server with the SSL option will result in a connection failure between the client and the server. The firewall configuration server authenticates a client using the client’s digital certificate and the Secure Sockets Layer (SSL) protocol. After a user supplies the digital certificate, the firewall configuration server checks that the certificate is valid. For more information, see *z/OS SecureWay Security Server Firewall Technologies*, SC24-5922.

A Resource Access Control Facility (RACF) key ring is used to store certificates that will be used by the local key server. This includes the local key server's own certificates, all CA certificates in the CA hierarchies that signed the local key server's certificates, and all CA certificates in the CA hierarchies used to sign a remote key server's certificate. It is expected that the remote key server's certificate will generally be signed by the same CA hierarchy that signed the certificate being used by the local key server. Certificates of remote key servers are not stored on the RACF key ring. The local key server will only require certificates if key policies supporting RSA Signature mode of authentication have been specified.

The Firewall Technologies contained within z/OS Version 1 Release 2 now allow the Configuration server's certificate to reside on a System Authorization Facility (SAF/RACF) key ring. Refer to *z/OS V1R2.0 SecureWay Security Server RACF Command Language Reference*, SA22-7687 for a complete description of the facilities and authorizations needed to create and modify digital certificates and key rings.

In order to address these enhancements, a configuration option was added to support the use of a SAF key ring by the Configuration server. An option was also added to indicate which certificate should be used by the Configuration server. This last option applies to both an SAF key ring or System SSL key file. By using a SAF key ring, you will get more security in the sense that:

- A stash file is no longer needed to save the password of a key file
- Access to the private key is controlled using External Security Managers (ESMs), like RACF, rather than UNIX file permissions and passwords
- The private key can be stored in the Integrated Cryptographic Service Facility (ICSF)

Configuration options for Firewall servers are defined using the **daemonopt** keyword on the **fwdaemon cmd=change** command. Two new options have been defined for the configuration server; they are:

-k with the name of the keyring, for certificates stored in an ESM

-l the certificate's label

Example 14-1 shows samples of possible configuration options.

Example 14-1 Configuring the server with SSL option

```
fwdaemon cmd=change daemon=CFGSRV daemonopts="-f /somedirectory/key.kdb -p 1014 -l
cfgcert"

fwdaemon cmd=change daemon=CFGSRV daemonopts="-k firewall key ring -l cfgcert"
```

Note:

1. **FWKERN** must be the owner of the key ring.
2. The **-k** an **-f** options are mutually exclusive.
3. If **-l** is not specified, the certificate marked as the default will be used.
4. The **-l** option may be used with either **-k** or **-f** options.

For more information about setting these parameters, see *z/OS SecureWay Security Server Firewall Technologies*, SC24-5922.

14.3 DNS configuration support removal

The Domain Name System (DNS) configuration support was removed. In z/OS Version 1 Release 2, Firewall Technologies no longer provides a command to support the configuration to the Bind 4.9.3 DNS server. The Bind 9.0 DNS server changes the format of DNS configuration files. In z/OS Version 1 Release 2, the Firewall Technologies will also not provide a command to support the configuration of the Bind 9.0 DNS server. The Communication server will ship a tool called dnsmigrate that will convert a Bind 4.9.3 named.boot file to a Bind 9.0 named.conf file. The **dnsmigrate** command is a migration aid that will convert ***named.boot*** files for the BIND.4.9.3 mode, into ***named.conf*** files suitable for the BIND.9 mode. The dnsmigrate command may only be run from the z/OS UNIX shell.

Information on configuring the DNS is found in *z/OS: Communications Server: IP Configuration Reference*, SC31-8776. Additional information may be found in *z/OS: Communications Server: IP Configuration Guide*, SC31-8775.

14.4 ISAKMP server enhancements

With the ISAKMP server, z/OS provides the capability to dynamically establish VPNs, negotiate VPN attributes, and dynamically manage VPN encryption keys. The ISAKMP Server offers a more automated, dynamic alternative to manual VPNs. Managing many manual VPNs can be tim-consuming and error-prone.

The ISAKMP server also automates the most important part of the VPN process: the secure and frequent exchange of encryption keys. The following improvements were made on the ISAKMP server on z/OS Version 1 Release 2:

- Commit bit
- VIPA support

14.4.1 Initializing security associations with ISAKMP and IKE

This section outlines how ISAKMP protocols initially establish security associations (SAs) and exchange keys between two systems that want to communicate securely.

The Internet Security Association and Key Management Protocol (ISAKMP) defines a framework for dynamically establishing security associations and cryptographic keys in an Internet environment. This framework defines a set of message flows and message formats. The message flows in ISAKMP are called *exchanges*. The message formats in ISAKMP are called *payloads*. ISAKMP defines a generic payload for key exchange information. This allows the ISAKMP protocol to manage cryptographic keys independent of the key exchange protocol that is used to generate them.

ISAKMP defers the interpretation of the key exchange payload to individual key exchange protocols. Internet Key Exchange (IKE) is such a protocol. IKE augments the ISAKMP protocol to facilitate the creation of authenticated keying material. IKE defines how keying material is generated. The exchanges defined by ISAKMP require authentication to take place, but they do not specify how authentication is to be performed. IKE defines how authentication is to be performed.

ISAKMP defines two phases of negotiation: Phase 1 and Phase 2. Both of these phases are also applicable to the IKE protocol.

- In Phase 1, two ISAKMP servers agree on how to protect traffic between themselves. This agreement results in the creation of an ISAKMP security association.

- In Phase 2, security associations (SAs) for other security protocols are established (for example, AH or ESP). Negotiations during each phase are accomplished by using an ISAKMP-defined exchange, or by an exchange specific to a key exchange protocol.

Phase 1

IKE supports two types of Phase 1 exchanges: main mode and aggressive mode. Both of these exchange modes are based on exchanges defined by ISAKMP. Main mode is an implementation of ISAKMP's Identity Protect Exchange. Aggressive mode is an implementation of ISAKMP's Aggressive exchange.

IKE defines four techniques for authentication of Phase 1 exchanges. These techniques are:

- Pre-shared key
- Signature-based
- Public key encryption
- Revised public key encryption

Note: Of these techniques, pre-shared key and signature-based using RSA signatures are supported by the z/OS Firewall Technologies ISAKMPD server.

For more details about Phase 1, see *SecureWay Security Server Firewall Technologies.*

Phase 2

IKE supports one type of Phase 2 exchange - quick mode. Quick mode is an IKE-specific exchange. It is not based on an ISAKMP-defined exchange.

Quick mode exchanges are bound to a specific Phase 1 exchange. This is accomplished by encrypting a hash of each quick mode message with a cryptographic key derived during the Phase 1 exchange. No explicit authentication of the identities involved in a Phase 2 exchange is performed.

For more details about Phase 2, see *SecureWay Security Server Firewall Technologies.*

Quick mode with commit bit

The ISAKMP protocol defines a bit in the ISAKMP message header known as the commit bit. When the commit bit is turned on during a quick mode exchange, the responder should acknowledge the receipt of message 3. The responder does this by extending the quick mode exchange to include a fourth message.

Messages 3 and 4, shown in Figure 14-6 on page 242, are used to exchange information specific to the generation of a shared secret key. This information includes Diffie-Hellman public values and a randomly generated value called a nonce. The initiator sends his Diffie-Hellman public value (i.e. g**x mod n) and a nonce in message 3. The responder sends his Diffie-Hellman public value (i.e. g**y mod n) and a nonce in message 4. With this information, both the responder and initiator can independently generate the identical keying information.

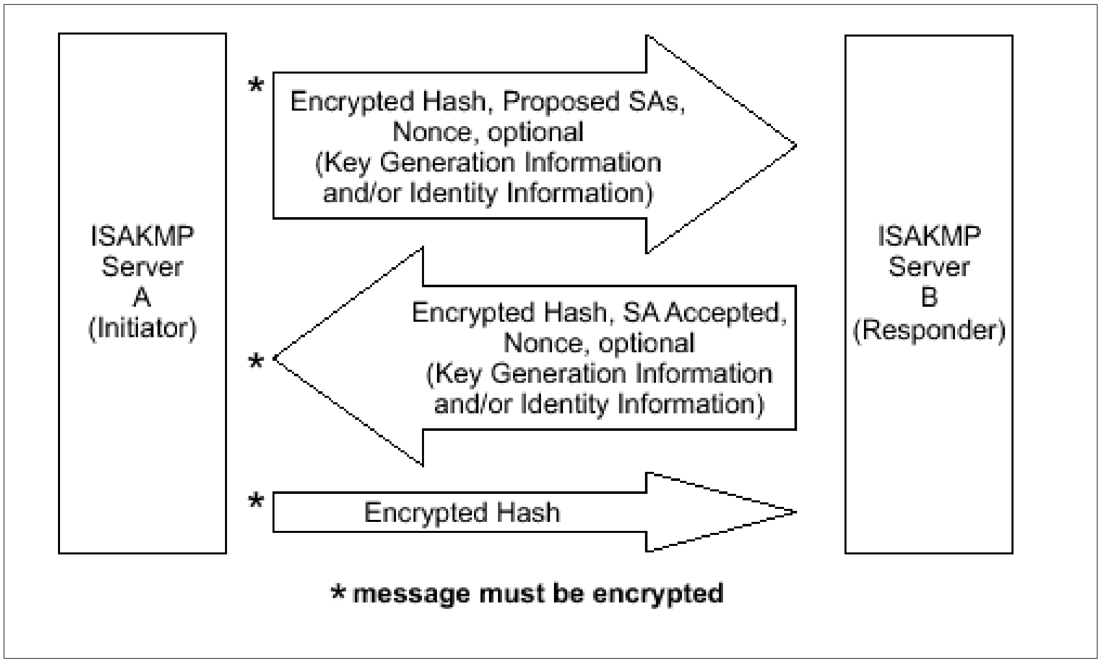

ISAKMP Server A (Initiator)
*
Encrypted Hash, Proposed SAs, Nonce, optional (Key Generation Information and/or Identity Information)
ISAKMP Server B (Responder)
*
Encrypted Hash, SA Accepted, Nonce, optional (Key Generation Information and/or Identity Information)
*
Encrypted Hash
* message must be encrypted

Figure 14-6 ISAKMP server messages 3 and 4

For more information about Phase 2, see *z/OS SecureWay Security Server Firewall Technologies*, SC24-5922.

Now the quick mode exchange is comprised of four messages, as shown in Figure 14-7 on page 243.

Commit bit

The commit bit is a bit defined in the header of messages used in the IKE protocol. It is actually defined in RFC 2407, which describes the Internet Security Association and Key Management Protocol (ISAKMP). IKE is based in ISAKMP. When the commit bit is set up to 1, it implies that commit bit logic should be applied.

The commit bit logic implemented in z/OS V1R2 is based on an IKE draft that was written after the 2409 IKE RFC. This draft differs significantly from the ISAKMP draft(2407); however, it seems most IPSec vendors implemented this IKE draft rather than RFC 2407.

The major advantages to the customer is increased interoperability and elimination of a potential disruptions of traffic flowing across a dynamic VPN. It helps to eliminate dropped IP packets that could occur during the process of negotiating new security associations.

In a normal quick mode exchange, the initiator can start using a newly negotiated SA immediately after sending message 3, as shown in Figure 14-7 on page 243. The responder will not start using the newly negotiated SA until it receives message 3. Message 3 is sent using UDP port 500. Since UDP is not a reliable protocol, it possible that the initiator will send message 3 and this message will never get processed by the responder. In this case, the responder will retransmit message 2 back to the initiator, causing the initiator to retransmit message 3. Unfortunately, during the period of the time between such retransmissions, the initiator might start using the SA to protect an IP packet. Any such packet would be discarded by the responder until it successfully processed message 3.

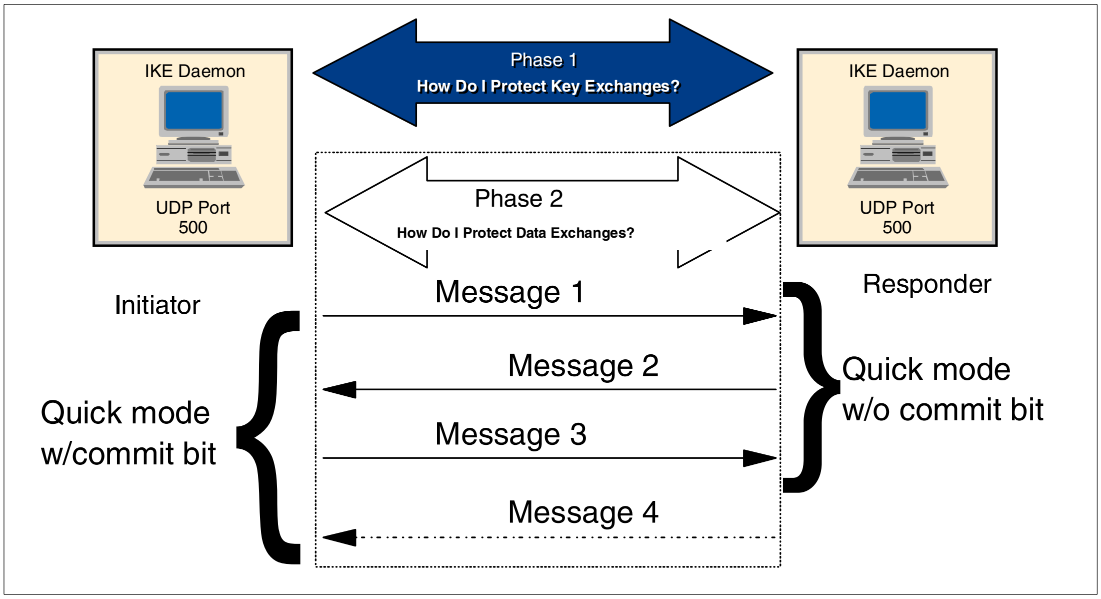


Figure 14-7 IPSec commit bit

In a quick mode exchange with commit processing, the initiator will defer the usage of a newly negotiated SA until one of the following events occur:

- The initiator receives a connected notify message
- The initiator receives an IP packet that was protected with the SA

The responder will continue to start using the newly negotiated SA when it receives message 3. This eliminates the window where one side might start using an SA before the other side knows that it is safe to use the SA.

On z/OS, an SA is considered to be in a pending state while the initiator is waiting for a connected notify message (that is, message 4). An SA will only be placed into a pending state if another SA that could be used to protect outbound traffic exits. An SA in pending state will remain in pending state until one of the following events occur:

- A connected notify is received
- A message protected by the SA is received
- The last usable SA expires

The commit bit logic is invoked internally by the ISAKMP daemon. No external configuration options to enable/disable commit bit processing are supported. When acting as a responder, the ISAKMP server will always turn on the commit bit in message 2. When acting as an initiator, it will always honor the setting of the commit bit in message 2.

The IKE draft that this support is based on indicates that either the initiator or the responder can initiate the use of the commit bit. Experience at IPSec conferences indicates that other venders only allow the responder to initiate the use of the commit bit. This is true with our support also.

The main intent of this line was to minimize the potential for dropping packets as the result of a refresh anomaly (that is, an SA being used before both peers know about it).

If no SAs are available to use when sending an outbound packet, the packet must be dropped. Given this, we concluded that on z/OS when a preexisting SA does not exist, that we will start to use a newly negotiated SA immediately (that is, after message 2). We concluded that it is better to send a packet that our peer might drop, rather than immediately dropping the packet.

The command **`fwdynconns`**, which configures dynamic connections, was enhanced to the following:

- **`fwdynconns cmd=listactive`**
  - display all non-expired security associations
  - display new commit bit-related states
    - pending
    - for use by inbound processing
    - for use by inbound and outbound processing

14.4.2 Virtual IP Address (VIPA) support

Traditionally, an IP address is associated with each end of a physical link (or each point of access to a shared-medium LAN), and the IP addresses are unique across the entire visible network, which can be the Internet or a closed intranet. The majority of IP hosts have a single point of attachment to the network, but some hosts (particularly large server hosts) have more than one link into the network. Because the virtual device exists only in software, it is always active and never experiences a physical failure. A VIPA has no single physical network attachment associated with it. Also, the TCP/IP stack does not maintain interface counters for VIPA interfaces (VIRTUAL links). A TCP/IP host with multiple points of attachment also has multiple IP addresses, one for each link.

Within the IP routing network, failure of any intermediate link or adapter disrupts end-user service only if there is not an alternate path through the routing network. Routers can route IP traffic around failures of intermediate links in such a way that the failures are not visible to the end applications or IP hosts. However, because an IP packet is routed based on ultimate destination IP address, if the adapter or link associated with the destination IP address fails, there is no way for the IP routing network to provide an alternate path to the stack and application.

Endpoint (source or destination) IP adapters and links thus constitute single points of failure. While this might be acceptable for a client host, where only a single user will be cut off from service, a server IP link might serve hundreds or thousands of clients, all of whose services would be disrupted by a failure of the server link.

The Virtual IP Address (VIPA) removes the adapter as a single point of failure by providing an IP address that is associated with a stack without associating it with a specific physical network attachment. Because the virtual device exists only in software, it is always active and never experiences a physical failure. A VIPA has no single physical network attachment associated with it.

To the routing network, a VIPA appears to be a host address indirectly attached to z/OS. When a packet with a VIPA destination reaches the stack, the IP layer recognizes the address and passes it to the protocol layer in the stack.

The failure of the physical interface can be extended to the failure of the TCP/IP address space, the entire z/OS, or for planned outages. A VIPA just needs to move to a backup stack, and the routes to the VIPA need to be updated. Then clients can transparently connect to the backup stack. This process is known as VIPA Takeover.

Support for VIPA Takeover

VIPA Takeover has been improved with the introduction of Dynamic Virtual IP Address (DVIPA) and Distributed Dynamic Virtual IP Address (Distributed DVIPA). The DVIPA function improves VIPA Takeover by allowing a systems programmer to plan for system outages and provide for backup systems to take over without operator intervention or external automation. The Distributed DVIPA function allows the connections for a single DVIPA to be serviced by applications on several stacks listed in the configuration statement (the distribution list). This adds the benefit of limiting the scope of an application or stack failure, while also providing enhanced workload balancing.

Dynamic VIPA support

The Communication Server has the concept of a dynamic VIPA (Virtual IP address). Dynamic VIPAs (DVIPAs) are defined in a VIPA dynamic block of the TCP/IP profile. A stack can be defined as the primary owner of a DVIPA or a backup owner of a DVIPA. If the stack that owns a DVIPA goes down, the DVIPA will be moved to a backup stack. The backup stack can be any stack in a sysplex (that is, it does not have to be on the same processor as the primary stack).

Prior to z/OS V1R2, when a DVIPA was moved to a firewall stack, the ISAKMP server did not create a UDP port 500 socket for this DVIPA until a request to activate a dynamic VPN was locally initiated using that DVIPA. In z/OS V1R2, the stack will inform the ISAKMP server when DVIPA is moved to a firewall stack. This allows the ISAKMP server to immediately create a UDP port 500 socket for newly arriving DVIPAs on a firewall stack and close sockets for DVIPAs being removed from a firewall stack. This allows a peer to request to activate a dynamic VPN without first requiring a locally activation or the ISAKMP server to be restarted.

When a DVIPA is moved from one firewall stack in a sysplex to another, the security associations (SA) associated with that DVIPA are not automatically reestablished on the target stack. They must be reestablished by issuing **fwdynconns** commands, on-demand activation, or by a peer initiation. No additional configuration is required to invoke this support.

14.5 Migration considerations

The z/OS V1R2 line items did not require any changes to the firewall configuration files, but that did not stop us from making changes anyway. A common suggestion relative to Dynamic VPN support have been to ship more predefined data policy and key policy objects. This allows new objects to exist in the configuration files by updating the fwmigrate utility.

The fwmigrate utility will add the newly predefined configuration objects. It will also issue an informational message indicating the DNS configuration files should be deleted when they are no longer being used (the utility could not just delete or move them because the might be still being used by the Communication Server’s Bind 4.9.3 DNS server).

14.5.1 Migration from z/OS Release 1

The **fwmigrate** command copies the current /etc/security DNS files to the /etc/security/fwbackup directory structure. The original files will not be deleted. If you choose to keep using the current DNS configuration, it will still be there. However, when this configuration needs to be changed next, or when you choose to move to the Communication Server's new Domain Name Server, the Communication Server's configuration procedures must be used.

Migration tasks

Additional pre-defined objects have been added to the following sample configuration files:

- fwahtran.cfg
- fwesptran.cfg
- fwdataprop.cfg
- fwdatapol.cfg
- fwdyntun.cfg
- fwkeytran.cfg
- fwkeyprop.cfg
- fwkeypol.cfg

The **fwmigrate** command adds the new pre-defined objects to the configuration files. If any of these files do not exist in the /etc/security directory, they must be copied from the /usr/lpp/fw/samples directory before executing the **fwmigrate** command.

On Windows-based systems, the z/OS V1 R2 GUI is installed in a different program folder than the OS/390 V2 R10 GUI. This makes it easier for users to use both GUIs from the same workstation.

15

z/OS cryptographic services

This chapter describes the following new elements of Integrated Cryptographic Services Facility (ICSF) for the z/OS cryptographic services available with z/OS Version 1 Release 2:

- Usability enhancements
- Customer-written UDX support
- ICC financial service support
- PKDS reencipher support
- PKA encrypt
- SSl performance enhancements

15.1 Integrated Cryptographic Services Facility

Integrated Cryptographic Services Facility is a software element of z/OS that works with the hardware cryptographic feature and the security server, Resource Access Control Facility (RACF), to provide secure, high-speed cryptographic services in the z/OS environment. ICSF provides the application programming interfaces by which applications request the cryptographic services. The cryptographic feature is secure, high-speed hardware that performs the actual cryptographic functions. The cryptographic feature available to your applications depends on the server or processor hardware. A number of changes have been made to enhance ICSF installation and initialization. These changes are described in the following sections.

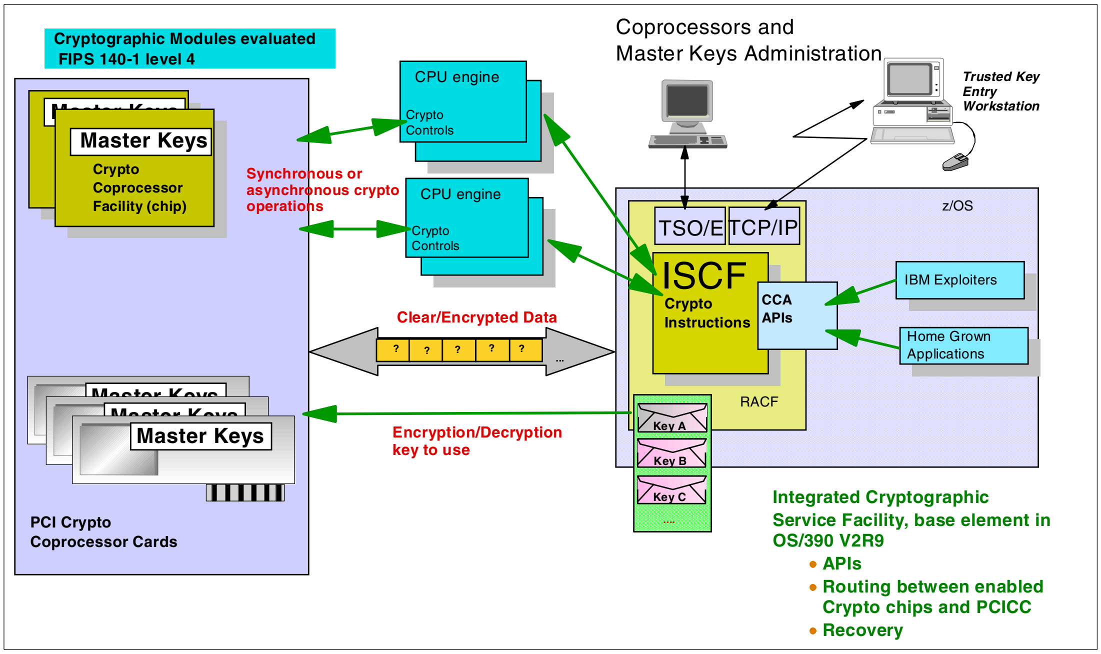

Cryptographic Modules evaluated FIPS 140-1 level 4
Master Keys
Crypto Coprocessor Facility (chip)
Synchronous or asynchronous crypto operations
CPU engine
Crypto Controls
CPU engine
Crypto Controls
Coprocessors and Master Keys Administration
Trusted Key Entry Workstation
z/OS
TSO/E
TCP/IP
ISCF
Crypto Instructions
CCA APIs
IBM Exploiters
Home Grown Applications
Clear/Encrypted Data
? ? ? ? ? ...
Master Keys
PCI Crypto Coprocessor Cards
Encryption/Decryption key to use
RACF
Key A
Key B
Key C
....
Integrated Cryptographic Service Facility, base element in OS/390 V2R9
• APIs
• Routing between enabled Crypto chips and PCICC
• Recovery

Figure 15-1 The S/390 integrated Cryptographic Coprocessors

15.1.1 Installing ICSF

Follow these steps to install ICSF:

1. Customize SYS1.PARMLIB.
2. Create the Cryptographic Key Data Set (CKDS).
3. Create the installation options data set.
4. Create the startup procedure.
5. Provide access to the ICSF panels.
6. Start ICSF for the first time.

For more detailed information about these steps, refer to *z/OS Integrated Cryptographic Service Facility System Programmer's Guide*, SA22-7520 and *z/OS Integrated Cryptographic Service Facility Administrator Guide*, SA22-7521.

15.1.2 New enhancements for V1R2

The following new elements that enhance the ICSF installation and initialization for the z/OS Cryptographic Services are available with z/OS Version 1 Release 2:

DOMAIN parameter

The DOMAIN parameter specifies the number of the domain that you want to use to start ICSF. You can specify only one domain in the options data set. Beginning with z/OS V1R2, DOMAIN is an optional parameter. The DOMAIN parameter is only required if more than one domain is specified as the usage domain on the PR/SM panels or if running in native mode. If specified in the options data set, it will be used and it must be one of the usage domains for the LPAR.

If DOMAIN is not specified in the options data set, ICSF determines which domains are available in this LPAR. If only one domain is defined for the LPAR, ICSF will use it. If more than one is available, ICSF issues error message CSFM409E.

The cryptographic processors support multiple sets of master key registers, which the specific domain values identify. The Cryptographic Coprocessor feature has a master key register for the DES master key, the auxiliary DES master key, the signature master key and the key management master key. The auxiliary DES master key register may contain either the new or old DES master key. On the PCI Cryptographic Coprocessor, each domain has a master key register for the current, new, and old SYM-MK and ASYM-MK. If you run ICSF in compatibility or coexistence mode, you cannot change the domain number without re-IPLing the system. A re-IPL ensures that a program does not access a cryptographic service with a key that is encrypted under a different master key. If you are certain that no cryptographic applications are still running, you can do the following:

- Stop ICSF.
- Start ICSF in COMPAT(NO) mode with a different domain number.
- Stop ICSF.
- Start ICSF in compatibility or coexistence mode with a different domain.

MAXLEN(n)

This parameter defines the maximum length of characters in a text string, including any necessary padding, for some callable service requests. For example, this option defines the maximum length of the text the encipher service encrypts for each call. Specify *n* as a decimal value from 1024 through 2147483647. If you do not specify the MAXLEN option, the default value is MAXLEN(65535). Beginning with z/OS V1R2, the MAXLEN parameter may still be specified in the options data set, but only the value limit value will be enforced (2147483647). If a value greater than this is specified, an error will result and ICSF will not start. Now, callable service will not check the text length value against the value in the MAXLEN parameter.

Note: Beginning with z/OS V1R2, MAXLEN is no longer displayed on the Installation Option Display panel.

15.2 The pass phrase initialization utility

The pass phrase initialization utility allows the casual user of ICSF to install the necessary master keys on both the Cryptographic Coprocessor features and the PCI Cryptographic Coprocessors, and initialize the CKDS with a minimal effort. This section describes how to use this utility to get up and running quickly.

15.2.1 Pass phrase initialization

The pass phrase initialization utility can be used to initialize the system and to initialize PCI Cryptographic Coprocessors that are brought online after system initialization. To use this utility, special secure mode must be enabled, and all master key registers must be empty. Note that you cannot use this utility to change master keys. To change master keys, you need to use either the clear master key entry panels or the TKE workstation. Since the same pass phrase will always produce the same master key values, you should secure the pass phrase in a safe place.

Before running the pass phrase initialization utility

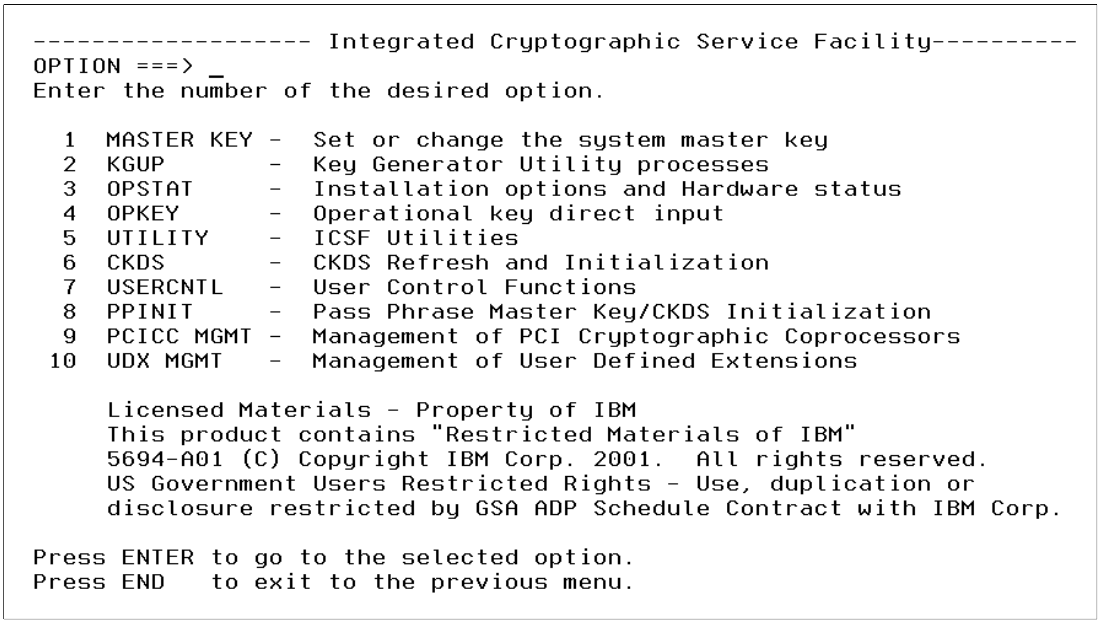
Before you run the pass phrase initialization utility for the first time, you must initialize ICSF by following these steps:

1. Install the ICSF program product according to the instructions in *z/OS Planning for Installation*, GA22-7504 and the ICSF Program Directory.
2. Create an empty CKDS.
3. Create an empty PKDS.
4. Create an installation options data set.
5. Create an ICSF startup procedure.
6. Start ICSF.
7. Access the ICSF panels.

These steps are described in *z/OS Integrated Cryptographic Service Facility System Programmer's Guide*, SA22-7520.

Running the pass phrase initialization utility

After you start ICSF, you can use the ICSF panels to run the pass phrase initialization utility. When you access the ICSF panels, the primary menu panel appears, as shown in Figure 15-2.

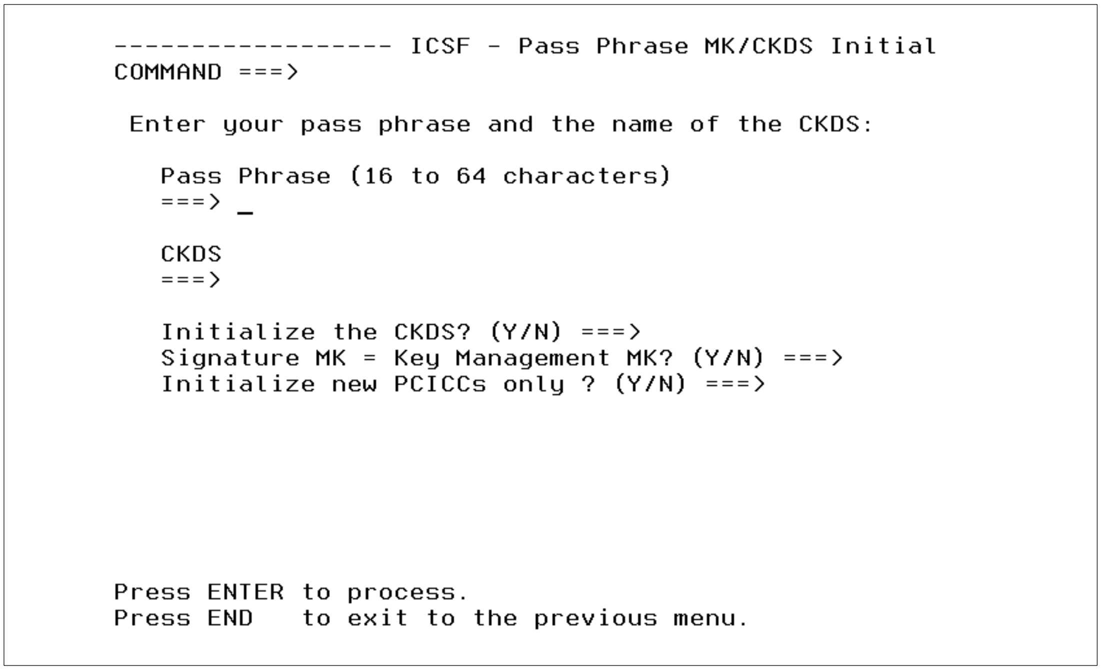


```
------------------- Integrated Cryptographic Service Facility----------
OPTION ===> _
Enter the number of the desired option.

  1  MASTER KEY -  Set or change the system master key
  2  KGUP       -  Key Generator Utility processes
  3  OPSTAT     -  Installation options and Hardware status
  4  OPKEY      -  Operational key direct input
  5  UTILITY    -  ICSF Utilities
  6  CKDS       -  CKDS Refresh and Initialization
  7  USERCNTL   -  User Control Functions
  8  PPINIT     -  Pass Phrase Master Key/CKDS Initialization
  9  PCICC MGMT -  Management of PCI Cryptographic Coprocessors
 10  UDX MGMT   -  Management of User Defined Extensions

     Licensed Materials - Property of IBM
     This product contains "Restricted Materials of IBM"
     5694-A01 (C) Copyright IBM Corp. 2001.  All rights reserved.
     US Government Users Restricted Rights - Use, duplication or
     disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 15-2 Selecting the pass phrase initialization option on the ICSF primary menu panel

1. Select option 8, `PPINIT`, and press Enter to begin the pass phrase initialization utility.The pass phrase is case-sensitive and should be chosen according to the following rules:
    a. It can contain a minimum of 16 and a maximum of 64 characters.
    b. It can include any characters in the EBCDIC character set.
    c. It can contain imbedded blanks, but leading and trailing blanks are truncated.

    The pass phrase MK/CKDS initialization panel appears; see Figure 15-3.



```
------------------ ICSF - Pass Phrase MK/CKDS Initial
COMMAND ===>

 Enter your pass phrase and the name of the CKDS:

   Pass Phrase (16 to 64 characters)
   ===> _

   CKDS
   ===>

   Initialize the CKDS? (Y/N) ===>
   Signature MK = Key Management MK? (Y/N) ===>
   Initialize new PCICCs only ? (Y/N) ===>





Press ENTER to process.
Press END   to exit to the previous menu.
```

Figure 15-3 ICSF pass phrase MK/CKDS initialization panel

2. Type the pass phrase and the data set name in the spaces provided. The CKDS name must be a valid MVS data set.
3. Answer the “Initialize the CKDS?” question by typing your response in the space following the question.
    a. If the CKDS has not been initialized, type Y. If you select Y, the CKDS name must refer to a valid, uninitialized CKDS.
    b. If this is an existing CKDS, type N. If you select N, the CKDS must have already been initialized with the pass phrase initialization utility and the identical pass phrase. ICSF checks and refreshes the existing CKDS.
4. Answer the “Signature MK = Key Management MK?” question by typing your response in the space following the question.
    a. If you have a new system with PCI Cryptographic Coprocessors installed, type Y. The signature master key and the key management master key will have the same value as the ASYM master key on the PCI Cryptographic Coprocessors. This increases the flexibility in routing services among the Cryptographic Coprocessors.
    b. If you have previously used pass phrase initialization and you have PKA key tokens that are encrypted under a key management master that cannot be recreated, type N. Press Enter to run the utility. For details of these calculations, refer to “Pass Phrase Initialization Master Key Calculations” in *z/OS Integrated Cryptographic Service Facility Administrator Guide*, SA22-7521.

5. When the utility has completed successfully, press End to return to the primary menu.

Adding PCICC after CCF initialization

The pass phrase initialization utility can be used to initialize PCI Cryptographic Coprocessors after system initialization. The procedure is:

- Disable PKA callable services.
- Reset NMK, SMK, and KMMK registers on the CCF.

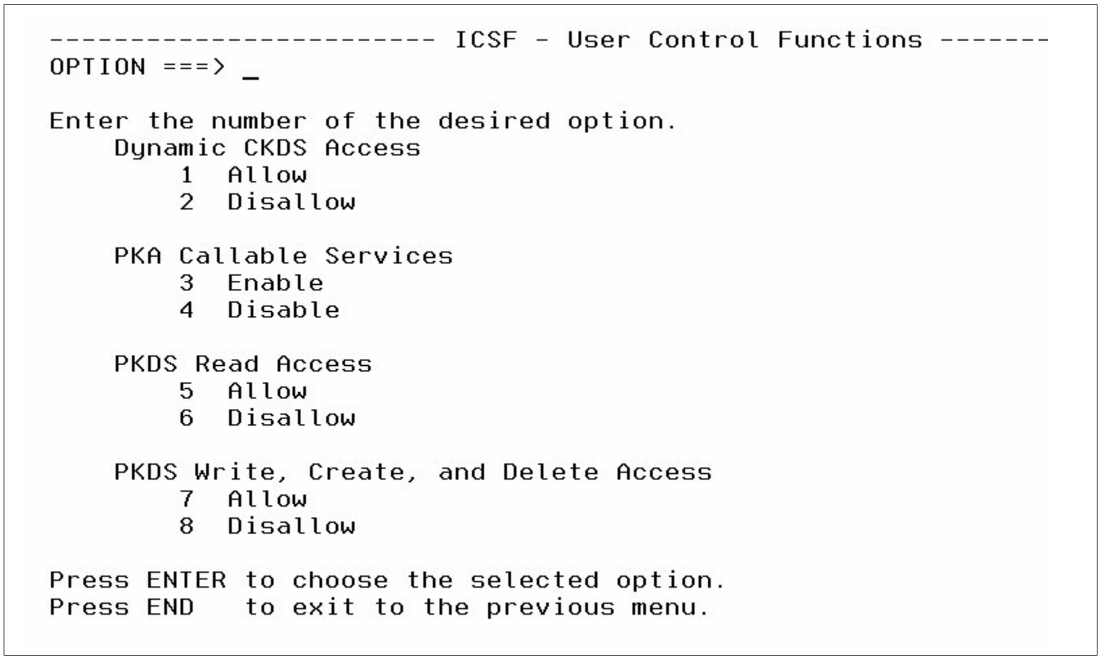
> **Note:** The NMK may not need to be reset. You only need to reset the NMK if an OMK exists or if you loaded an NMK through Clear Master Key Entry or the TKE workstation.

Run pass phrase initialization utility

The step-by-step procedure is:

1. Access the user control functions by choosing option 7, `USERCNTL`, on the Primary Menu panel, as shown in the following Figure 15-2 on page 250.

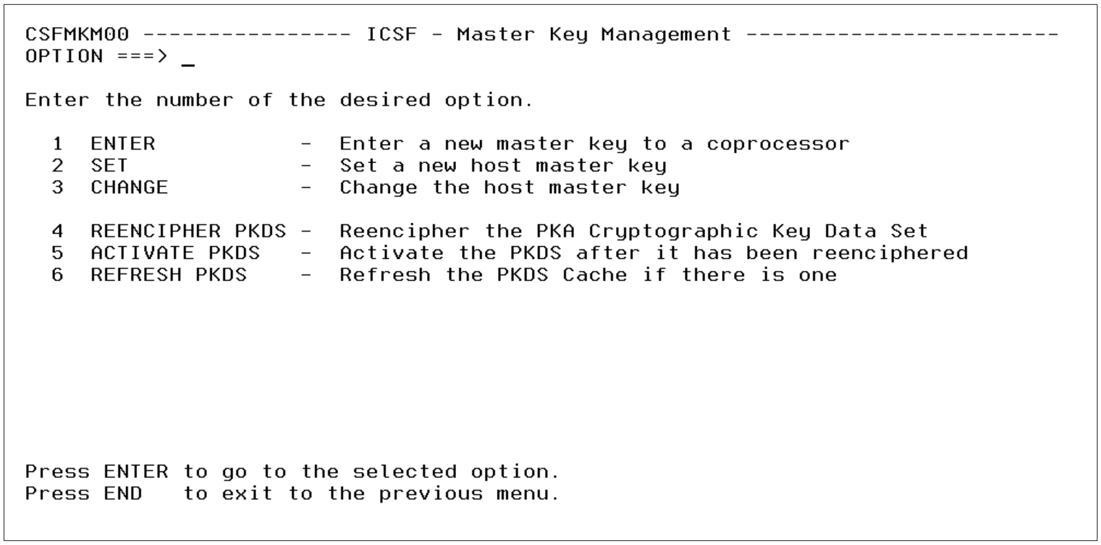
The User Control Function panel appears, as shown in Figure 15-4.



```
------------------------ ICSF - User Control Functions -------
OPTION ===> _

Enter the number of the desired option.
    Dynamic CKDS Access
        1  Allow
        2  Disallow

    PKA Callable Services
        3  Enable
        4  Disable

    PKDS Read Access
        5  Allow
        6  Disallow

    PKDS Write, Create, and Delete Access
        7  Allow
        8  Disallow

Press ENTER to choose the selected option.
Press END   to exit to the previous menu.
```

Figure 15-4 Enabling and Disabling the PKA callable services

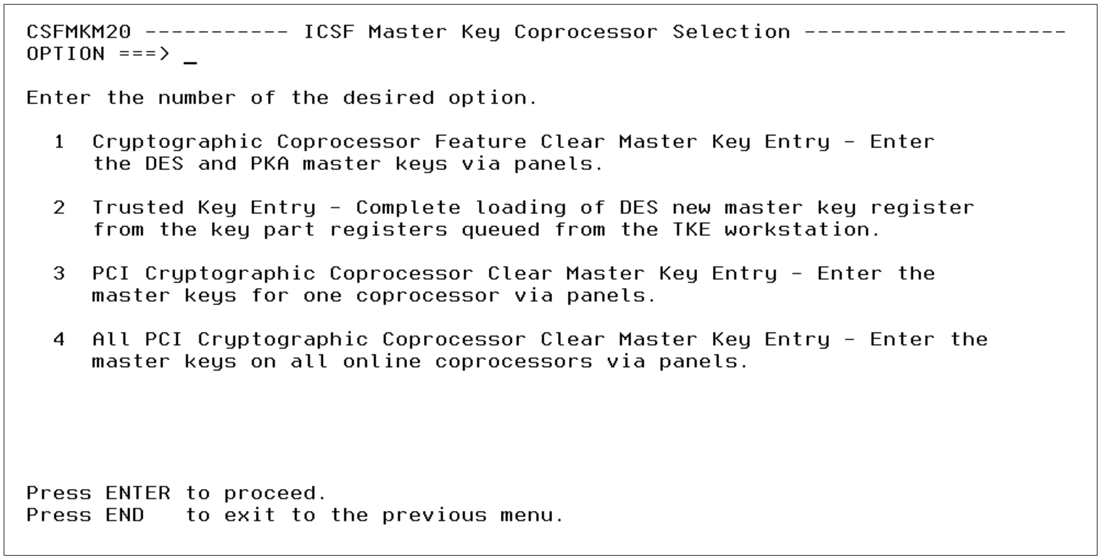
2. Enter an option and press Enter. To disable the PKA callable services, select option 4, `Disable`.
3. Press PF3 on the USERCNTL panel and the primary menu panel once again appears.
4. This time select option 1, `MASTER KEY`, and press Enter.
5. The first Master Key Management panel appears, as shown in Figure 15-5 on page 253. Select option 1 and press Enter.



```
CSFMKM00 ---------------- ICSF - Master Key Management ------------------------
OPTION ===> _

Enter the number of the desired option.

  1  ENTER           -  Enter a new master key to a coprocessor
  2  SET             -  Set a new host master key
  3  CHANGE          -  Change the host master key

  4  REENCIPHER PKDS -  Reencipher the PKA Cryptographic Key Data Set
  5  ACTIVATE PKDS   -  Activate the PKDS after it has been reenciphered
  6  REFRESH PKDS    -  Refresh the PKDS Cache if there is one







Press ENTER to go to the selected option.
Press END   to exit to the previous menu.
```

Figure 15-5 Selecting the Enter option on the Master Key management panel

6. Another Master Key Management panel appears, as shown in Figure 15-6. Enter 1, `Cryptographic Coprocessor Feature Clear Master Key Entry`, and press Enter.

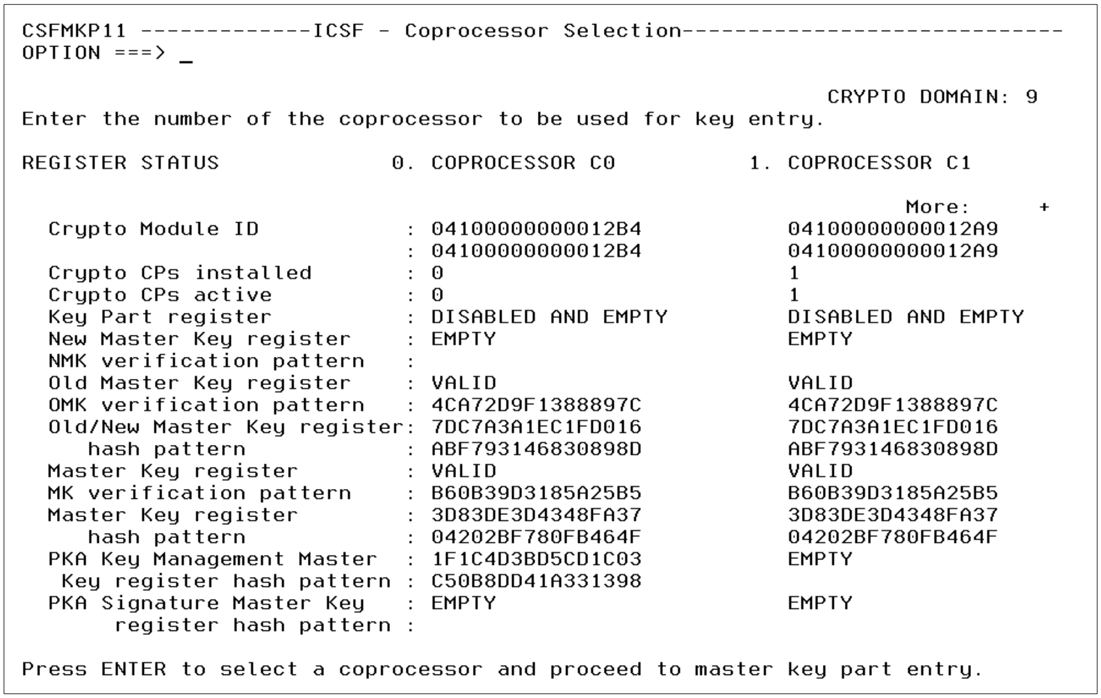


```
CSFMKM20 ----------- ICSF Master Key Coprocessor Selection --------------------
OPTION ===> _

Enter the number of the desired option.

  1  Cryptographic Coprocessor Feature Clear Master Key Entry - Enter
     the DES and PKA master keys via panels.

  2  Trusted Key Entry - Complete loading of DES new master key register
     from the key part registers queued from the TKE workstation.

  3  PCI Cryptographic Coprocessor Clear Master Key Entry - Enter the
     master keys for one coprocessor via panels.

  4  All PCI Cryptographic Coprocessor Clear Master Key Entry - Enter the
     master keys on all online coprocessors via panels.




Press ENTER to proceed.
Press END   to exit to the previous menu.
```

Figure 15-6 Selecting the CCF Clear Master Key Entry option on the Master Key management panel

7. The ICSF Coprocessor Selection Panel appears. Select the Coprocessor for master key entry by typing the Coprocessor number on the OPTION line and pressing Enter. See Figure 15-7 on page 254.

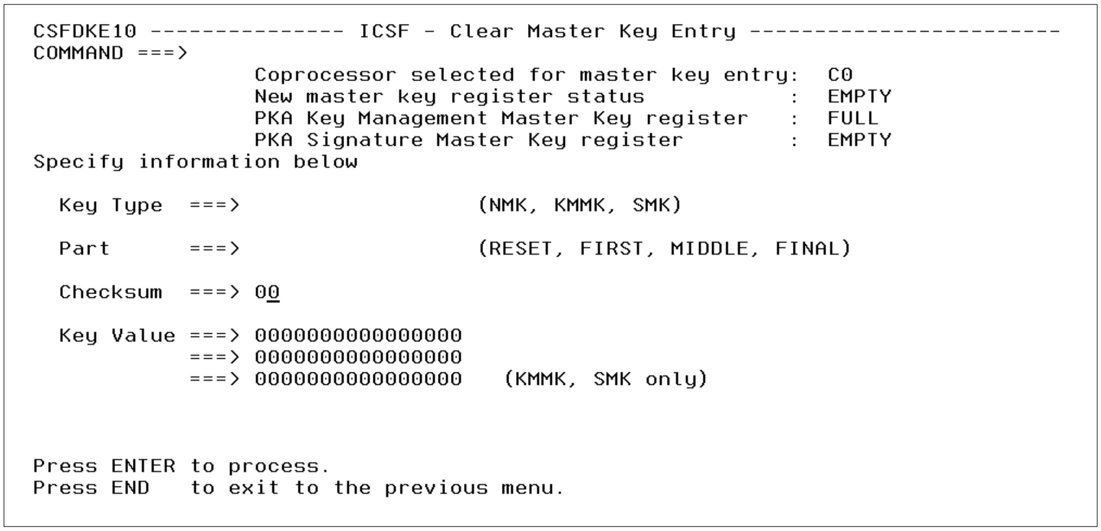
> **Note:** If you have only one Coprocessor installed, or if there is only one Coprocessor defined to this LP, this panel will only show one Coprocessor.



```
CSFMKP11 -------------ICSF - Coprocessor Selection-----------------------------
OPTION ===> _
                                                             CRYPTO DOMAIN: 9
Enter the number of the coprocessor to be used for key entry.

REGISTER STATUS            0. COPROCESSOR C0          1. COPROCESSOR C1

                                                             More:     +
  Crypto Module ID          : 04100000000012B4           04100000000012A9
                            : 04100000000012B4           04100000000012A9
  Crypto CPs installed      : 0                          1
  Crypto CPs active         : 0                          1
  Key Part register         : DISABLED AND EMPTY         DISABLED AND EMPTY
  New Master Key register   : EMPTY                      EMPTY
  NMK verification pattern  :
  Old Master Key register   : VALID                      VALID
  OMK verification pattern  : 4CA72D9F1388897C           4CA72D9F1388897C
  Old/New Master Key register: 7DC7A3A1EC1FD016           7DC7A3A1EC1FD016
     hash pattern           : ABF793146830898D           ABF793146830898D
  Master Key register       : VALID                      VALID
  MK verification pattern   : B60B39D3185A25B5           B60B39D3185A25B5
  Master Key register       : 3D83DE3D4348FA37           3D83DE3D4348FA37
     hash pattern           : 04202BF780FB464F           04202BF780FB464F
  PKA Key Management Master : 1F1C4D3BD5CD1C03           EMPTY
   Key register hash pattern : C50B8DD41A331398
  PKA Signature Master Key  : EMPTY                      EMPTY
       register hash pattern :

Press ENTER to select a coprocessor and proceed to master key part entry.
```

Figure 15-7 Coprocessor selection panel

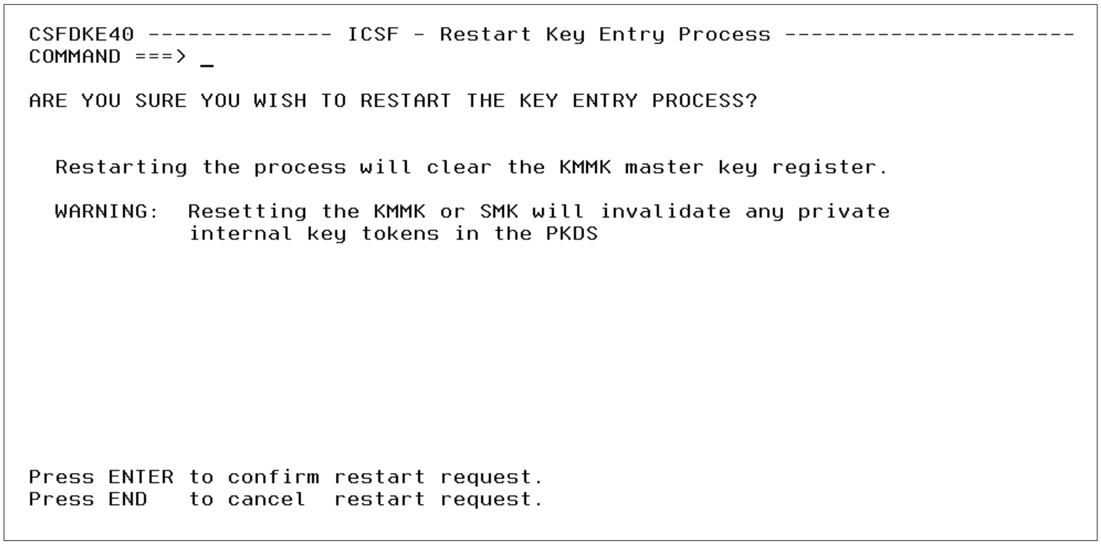
8. The Clear Master Key Entry panel appears. You need to RESET to clear the contents of the registers before you can set a new key value.



```
CSFDKE10 --------------- ICSF - Clear Master Key Entry ------------------------
COMMAND ===>
                 Coprocessor selected for master key entry:  C0
                 New master key register status           :  EMPTY
                 PKA Key Management Master Key register   :  FULL
                 PKA Signature Master Key register        :  EMPTY
Specify information below

  Key Type  ===>                  (NMK, KMMK, SMK)

  Part      ===>                  (RESET, FIRST, MIDDLE, FINAL)

  Checksum  ===> 00

  Key Value ===> 0000000000000000
            ===> 0000000000000000
            ===> 0000000000000000   (KMMK, SMK only)



Press ENTER to process.
Press END   to exit to the previous menu.
```

Figure 15-8 The Clear Master Key entry panel to reset registers

9. When you select `RESET`, the Restart Key Entry Process panel is displayed; see Figure 15-9 on page 255. This panel confirms your request to restart the key entry process. Press Enter.



```
CSFDKE40 -------------- ICSF - Restart Key Entry Process ----------------------
COMMAND ===> _

ARE YOU SURE YOU WISH TO RESTART THE KEY ENTRY PROCESS?


  Restarting the process will clear the KMMK master key register.

  WARNING:  Resetting the KMMK or SMK will invalidate any private
            internal key tokens in the PKDS









Press ENTER to confirm restart request.
Press END   to cancel  restart request.
```

Figure 15-9 Confirm restart request panel

10. You must also reset the SMK and NMK.

> **Note:** You need to reset the NMK if an OMK exists or if you loaded an NMK through Clear Master Key Entry or the TKE workstation.

11. If you have two Cryptographic Coprocessors, you must repeat the process for the second Cryptographic Coprocessor. Press PF3 and you should be at Figure 15-6 on page 253. So, repeat the process.
12. Once the CCFs have been cleared, run the Pass Phrase Initialization Utility. Access the primary menu panel. (See Figure 15-2 on page 250.)
13. Select option 8, `PPINIT`, and press Enter to begin the pass phrase initialization utility.The Pass Phrase MK/CKDS Initialization panel appears. (See Figure 15-3 on page 251.)

> **Note:** You are reentering master keys after they have been cleared and must use the same pass phrase as when you originally entered the keys. You should have saved the pass phrase in a secure place after you entered the master keys previously.

14. Type the pass phrase and the data set name in the spaces provided. The CKDS name must be a valid MVS data set.
15. Answer the “Initialize the CKDS?” question by typing your response in the space following the question. This is an existing CKDS, so you must type N. ICSF checks and refreshes the existing CKDS.
16. Answer the “Signature MK = Key Management MK?” question by typing your response in the space following the question.
    a. If you have a new system with PCI Cryptographic Coprocessors installed, type Y. The signature master key and the key management master key will have the same value as the ASYM master key on the PCI Cryptographic Coprocessors. This increases the flexibility in routing services among the Cryptographic Coprocessors.
    b. If you have previously used pass phrase initialization and you have PKA key tokens that are encrypted under a key management master that cannot be recreated, type N.
17. Press Enter to run the utility.

Messages on the bottom half of the panel display the progress of the utility.

18. When the utility has completed successfully, press End to return to the primary menu.

15.3 ABENDS - message replacement

ICSF writes messages to data sets and consoles. You can view some messages immediately as they appear on the console and you can view messages in data sets. This section describes which ABENDS have been eliminated and replaced with operator messages.

Message IEC161I has been eliminated during the first-time startup of ICSF. The following reason codes for ICSF/MVS X'18F' are being eliminated and will be replaced with operator messages:

- Reason Code X'3C' - replaced by message CSFM105E
- Reason Code X'48' - replaced by message CSFM120E
- Reason Code X'1B' - replaced by message CSFM410E
- Reason Code X'4B' - replaced by message CSFM107E
- Reason Code X'106'

If the CCC is all zeroes, abend X'18F' reason code 4A will occur. If the CCC does not exist, message CSFM113E will be displayed.

15.4 Customer-written UDX support

The Common Cryptographic Architecture (CCA) requires that security-sensitive functions are carried out in an environment where secret or private quantities can safely appear in the clear and where the design of the processing functions can not be altered by an adversary. A Coprocessor application program operates in such an environment. However, the confidentiality of secret or private quantities (for example, cryptographic keys or computational values) is also the responsibility of the application program design.

The CCA application operates as a request/response mechanism. Once initialized by CP/Q ++ as a result of a Coprocessor reset sequence, the CCA application within the Coprocessor waits for an external request. The application then performs the requested function and returns a response.

The application retains persistent data as a set of security-relevant data items (SRDI). The application stores SRDIs in RAM memory, with a backup copy retained in either battery-backed RAM (BBRAM) or (optionally) encrypted in flash memory.

The CCA verbs (callable services) that a host application can request are generally serviced, on a one-for-one basis, by a command processor portion of Coprocessor application code. A common infrastructure is employed to format a verb request, transport the request to the Coprocessor, dispatch the command processor, and return the reply to the host. Command processors and the top layer of CCA host code, security application program interface (SAPI), make extensive use of a set of common subroutines described in this redbook.

The code that implements a user-defined extension (UDX) to CCA can be separated into two distinct pieces. One (the “host piece”) runs as a DLL on the host. The other (the “Coprocessor piece”) is linked with a library containing the IBM CCA Coprocessor application modules and downloaded to the Coprocessor.

The UDX Development Toolkit for the IBM 4758 provides scaffold code, object modules, and header files that you can use to extend the IBM-developed CCA application program which employs the IBM 4758 PCI Cryptographic Coprocessor. You can use as much or as little of the CCA application function as required to meet your processing requirements.

Beginning with OS/390 V2 R10 ICSF, ICSF support is provided for UDX capabilities. UDX routines are developed by special contract with IBM and are only distributed to authorized customers.

The UDX function is invoked by an “installation-defined” or generic callable service. The callable service is defined in the Installation Options data set (UDX parameter) and the service stub is link-edited with the application. The application program calls the service stub, which accesses the UDX installation-defined service. There is a one-to-one correspondence between a specific generic service in ICSF and a specific UDX command processor in the PCI Cryptographic Coprocessor. The administrator, through TSO panels, performs UDX authorization processing on each PCI Cryptographic Coprocessor.

For more details see *IBM 4758 PCI Cryptographic Coprocessor CCA User Defined Extensions Reference and Guid*e, *Version 2:* 4758-002 and 4758-023

15.4.1 ICSF support

The ICSF supports customer development of UDX with a tool kit through an OEM support contract. UDX support allows you to request implementation of a customized cryptographic callable service. UDXs are ICSF functions developed for your installation with the help of IBM Global Services.You must define your routine to ICSF in the Installation Options data set. For more detailed information on the Installation Options data set and the new UDX keyword, see *z/OS Integrated Cryptographic Service Facility System Programmer’s Guide*, SA22-7520.

The generic service load module is loaded during ICSF startup. Use the ICSF panels to perform UDX authorization processing, as you can see in Figure 15-2 on page 250.

Once you have selected option 10, you can perform the following tasks:

- Display a list of UDX IDs of all authorized UDXs on a specific PCI Cryptographic Coprocessor
- Display a list of all PCI Cryptographic Coprocessors on which a specific UDX is authorized
- Authorize a UDX on any PCI Cryptographic Coprocessor in the system

A zSeries z900 with PCICC is required for customer-written UDX. For more details, see *z/OS Integrated Cryptographic Service Facility Administrator Guide*, SA22-7521*.*

15.5 ICC financial services

ICSF provides the following services for emerging integrated circuit card (ICC) standards to support smartcard financial applications:

- Secure Messaging for Keys (CSNBSKY) callable service recovers a clear key from a token, embeds it in the message and encrypts the message.

This callable service will encrypt a text block including a clear key value decrypted from an internal or external DES token. The text block is normally a ‘Value‘ field of a secure message TLV (Tag/Length/Value) element of a secure message. TLV is defined in ISO/IEC 7816-4. Processing for this service is routed to the PCI Cryptographic Coprocessor. Keys only appear in the clear within the secure boundary of the PCI Cryptographic Coprocessor, and never in host storage. The encryption mode may be either CBC or ECB.

- Secure Messaging for PINs (CSNBSPN) callable service recovers a PIN from an encrypted PIN block, optionally reformats the PIN block, embeds the PIN in a message and encrypts the message.

  This callable service will encrypt a text block including a clear PIN block recovered from an encrypted PIN block. The input PIN block will be reformatted if the block format in the *i*nput_PIN_profile is different than the block format in the output_PIN_profile. The clear PIN block will only be self-encrypted if the SELFENC keyword is specified in the rule_array. The text block is normally a ‘Value‘ field of a secure message TLV element of a secure message. TLV is defined in ISO/IEC 7816-4. Processing for this service is routed to the PCI Cryptographic Coprocessor. PINs only appear in the clear within the secure boundary of the PCI Cryptographic Coprocessor, and never in host storage.The encryption mode may be either CBC or ECB. See *z/OS Integrated Cryptographic Service Application Programmer’s Guide*, SA22-7522 for more details.

15.6 PKDS reencipher support

ICSF supports the ability to change the PKA master keys via:

- New TSO utility to reencipher the PKA Key Data Set
- New utility program CSFPUTIL
- PKA Key Token Change callable service

PKDS reencipher allows customers with PCI Cryptographic Coprocessors to change their master keys. The utility program CSFPUTIL allows customers to perform the same PKDS reencipher operations using a batch job. In addition, a new callable service, PKA Key Token Change (CSNDKTC), will allow customers to reencipher PKA key tokens to the current asymmetric-keys master key.

Support to REENCIPHER PKDS and ACTIVATE PKDS has been added to the Master Key Management panels and to the CSFPUTIL utility to reencipher the PKDS from the old asymmetric-keys master key to the current master key, and to activate the reenciphered PKDS.

PKDS Reencipher/Activate options are available to reencipher RSA and DSS internal tokens in the PKDS after the SMK/ASYM-MK keys are changed. PKA master keys may not be changed dynamically. There is no reencryption capability for keys enciphered under PKA master keys. If a PKA master key is changed, any internal tokens in the PKDS containing keys enciphered by that PKA master key must be recreated.

The PKA Key Token Change callable service changes PKA key tokens (RSA and DSS) from encipherment under the old PCI Cryptographic Coprocessor Asymmetric-Keys Master Key to encipherment under the current PCI Cryptographic Coprocessor Asymmetric-Keys Master Key. This service only changes Private Internal PKA Key Tokens. PKA private keys encrypted under the Key Management Master Key (KMMK) cannot be reenciphered using this service unless the KMMK has the same value as the Signature Master Key (SMK).

The PKA Key Token Change requests a PCI Cryptographic Coprocessor for processing. If no PCI Cryptographic Coprocessor is online, the request fails.

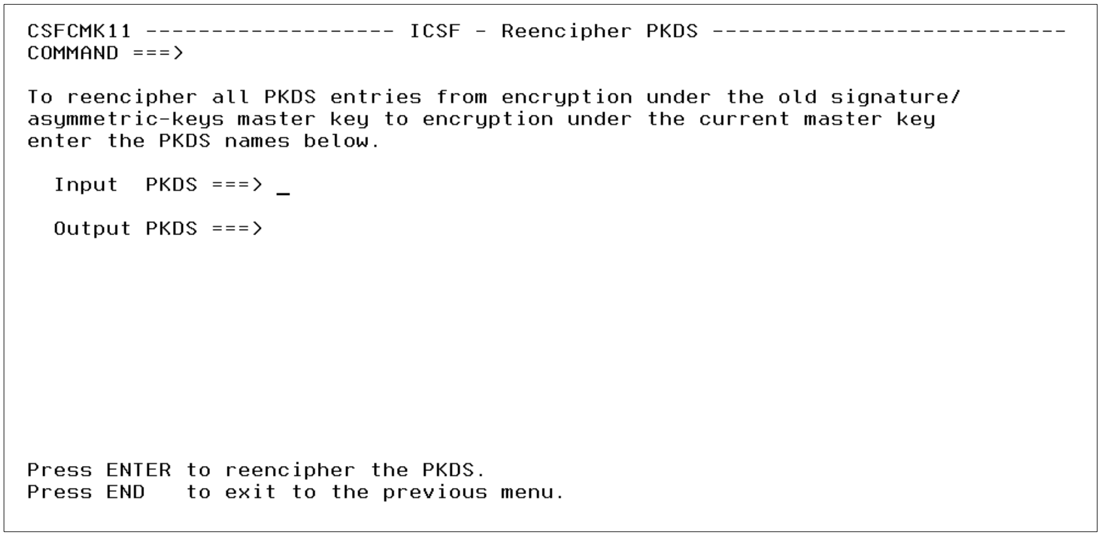
Reencipher the PKDS

1. Enter the keys part of the new SMK/ASYM-MKPKA master keys that you want to replace the currents keys. For detailed information about how to do this, see *z/OS Integrated Cryptographic Service Facility Administrator Guide*, SA22-7521.
2. Go to the primary panel of ICSF and select option 1, `MASTER KEY`, from the ICSF Primary menu, as shown in Figure 15-2 on page 250; press Enter.

   The Master Key Management panel appears.
3. Select option 4, `REENCIPHER PKDS`, on the Master Key Management panel, as shown in Figure 15-5 on page 253; press Enter.
4. The ICSF Reencipher panel appears, as shown in Figure 15-10. In the Input PKDS field, specify the name of the PKDS that you want ICSF to reencipher under the current signature/asymmetric-keys master key. In the Output PKDS field, specify the name of an empty VSAM data set. ICSF places the reenciphered keys in this data set.

   Note: The output data set should already exist, although it must be empty. For more information about defining a PKDS, see the *z/OS Integrated Cryptographic Service Facility System Programmer's Guide*, SA22-7520.

   Reenciphering the disk copy of the PKDS does not affect the in-storage copy of the PKDS. On this panel, you are working with only a disk copy of the PKDS.



```
CSFCMK11 ------------------- ICSF - Reencipher PKDS ---------------------------
COMMAND ===>

To reencipher all PKDS entries from encryption under the old signature/
asymmetric-keys master key to encryption under the current master key
enter the PKDS names below.

  Input  PKDS ===> _

  Output PKDS ===>










Press ENTER to reencipher the PKDS.
Press END   to exit to the previous menu.
```

Figure 15-10 CSF: Reencipher PKDS panel

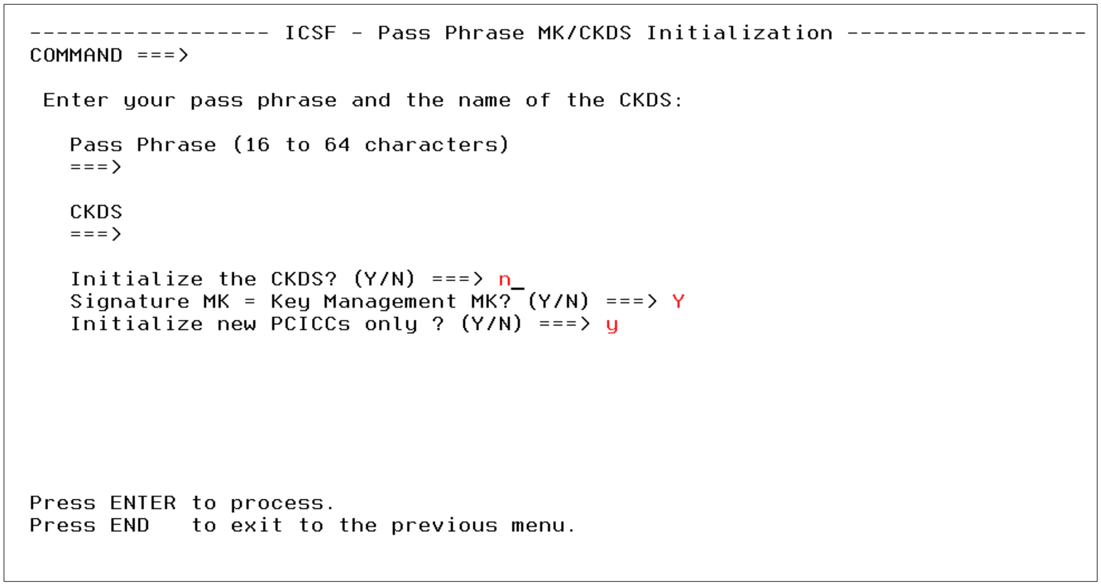
5. Press Enter to reencipher all PKDS entries in the input PKDS from encryption under the old signature/asymmetric master key to encryption under the current master key.

   The message "REENCIPHER SUCCESSFUL" appears on the top right of the panel if the reencipher succeeds.
6. If you have more than one PKDS on disk, specify the information and press Enter as many times as you need to reencipher all of them. Reencipher all your disk copies at this time. When you have reenciphered all the disk copies of the PKDS, you are ready to change the master key.

7. Return to the Master Key Management panel and select option 5, `ACTIVATE PKDS`, as show in Figure 15-5; press Enter.

   The ICSF - Activate PKA Cryptographic Key Data Set appears.

8. As show in Figure 15-11 on page 260, in the New PKDS field, enter the name of the PKDS that you want ICSF to use. The PKDS must have already been reenciphered under the current signature/asymmetric-keys master key. After you press Enter, the PKDS becomes active and PKA callable services are enabled.



```
------------------ ICSF - Pass Phrase MK/CKDS Initialization ------------------
COMMAND ===>

 Enter your pass phrase and the name of the CKDS:

   Pass Phrase (16 to 64 characters)
   ===>

   CKDS
   ===>

   Initialize the CKDS? (Y/N) ===> n_
   Signature MK = Key Management MK? (Y/N) ===> Y
   Initialize new PCICCs only ? (Y/N) ===> y





Press ENTER to process.
Press END   to exit to the previous menu.
```

Figure 15-11 ICSF-Activate PKA cryptographic key data set panel

15.7 PKA encrypt

PKA encrypt callable service supports the ZERO-PAD keyword. The input key value is padded with binary zero and encrypted under the RSA key. This callable service encrypts a supplied clear key value under an RSA public key. Currently, the supplied key can be formatted using the PKCS 1.2 or ZERO-PAD methods prior to encryption. The rule array keyword specifies the format of the key prior to encryption. This service routes requests to the Cryptographic Coprocessor feature unless the modulus bit length of the key specified in the PKA_key_identifier is greater than 1024 bits. PKA Encrypt callable service supports the ZERO-PAD keyword. The input key value is padded with binary zero and encrypted under the RSA key. See *z/OS Integrated Cryptographic Service Application Programmer's Guide*, SA22-7522 for more details.

15.8 SSL performance enhancements

The PCI Cryptographic Coprocessor encapsulates a 486-class processing subsystem within a tamper-sensing and tamper-responding environment where you can run security-sensitive processes. A multi-tasking control program gives you access to high-performance electronics for DES and public key algorithms. A cryptographic-quality random number generation facility and large, secure, persistent data storage complete the physical system.

Secure Web servers, and many other kinds of servers that want to protect data from eavesdropping during transmission, often use the Secure Socket Layer (SSL) protocol. The SSL protocol, developed by Netscape Development Corporation, provides communications privacy and prevents eavesdropping, tampering, or message forgery. Once the browser and the server have completed the handshake, they can encrypt the rest of their communication using a standard encryption algorithm such as DES or RC4. These algorithms are known as “ciphers,” and this technique is called “symmetric cryptography.” Symmetric cryptography uses a common key for both encrypting and decrypting data, and this cryptography is generally very fast. But as the browser and server may not have interacted before, they need some way of establishing that common key. This is achieved in the handshake by key-exchange techniques that are based on asymmetric encryption (public/private key cryptography), RSA or DSS. This is commonly based on the RSA algorithm and typically uses a 1024-bit RSA key-pair. ICSF provides callable services that support the RSA-encryption and RSA-decryption of PKCS 1.2-formatted symmetric key data to produce symmetric session keys.

PKA decrypt callable service

The PKA decrypt callable service uses the corresponding private RSA key to unwrap the RSA-encrypted key and deformat the key value. This service then returns the clear key value to the application.

PKA encrypt callable service

The PKA encrypt callable service encrypts a supplied clear key value under an RSA public key. Currently, the supplied key can be formatted using the PKCS 1.2 or ZERO-PAD methods prior to encryption.

These session keys can then be used to establish an SSL session between the sender and receiver. For more details, see *z/OS Integrated Cryptographic Service Facility System Programmer’s Guide*, SA22-7520.

PCI Cryptographic Accelerator

To enhance the performance for SSL applications, the PCI Cryptographic Accelerator is supported. The PCI Cryptographic Accelerator is a PCI card with five RSA engines for clear key operations. The RSA public-key cryptographic operations usually used to exchange the session key at the start of a connection are computationally intensive. The accelerator gives you a specified level of performance for handling the time-consuming RSA operations.

PKDS cache support

RSA and DSS public and private keys can be stored in the PKA key data set (PKDS), a VSAM data set.The PKDS is automatically initialized at ICSF startup. There are internal and external tokens in the PKDS. External tokens may be used irrespective of the PKA master keys. Internal tokens, however, can only be used if they are encrypted under the current PKA signature master key (SMK), the key management master key (KMMK), or the asymmetric-keys master key (ASYM-MK). For more details, see *z/OS Integrated Cryptographic Service Facility Administrator Guide*, SA22-7521*.*

To perform SSl applications, the keys stored in the PKDS are necessary. So, to increase the performance, a more effective access to PKDS keys was made functional. A PKDS cache was added. In this way, PKA tokens in the PKDS are cached in storage as they are used, increasing the throughput. The PKDS cache will benefit any application that uses the PKDS.

16

Security server Kerberos

In this chapter we describe briefly the Kerberos protocol, and discuss the new enhancements introduced in the z/OS Version 2 Release1.

This chapters includes the following topics:

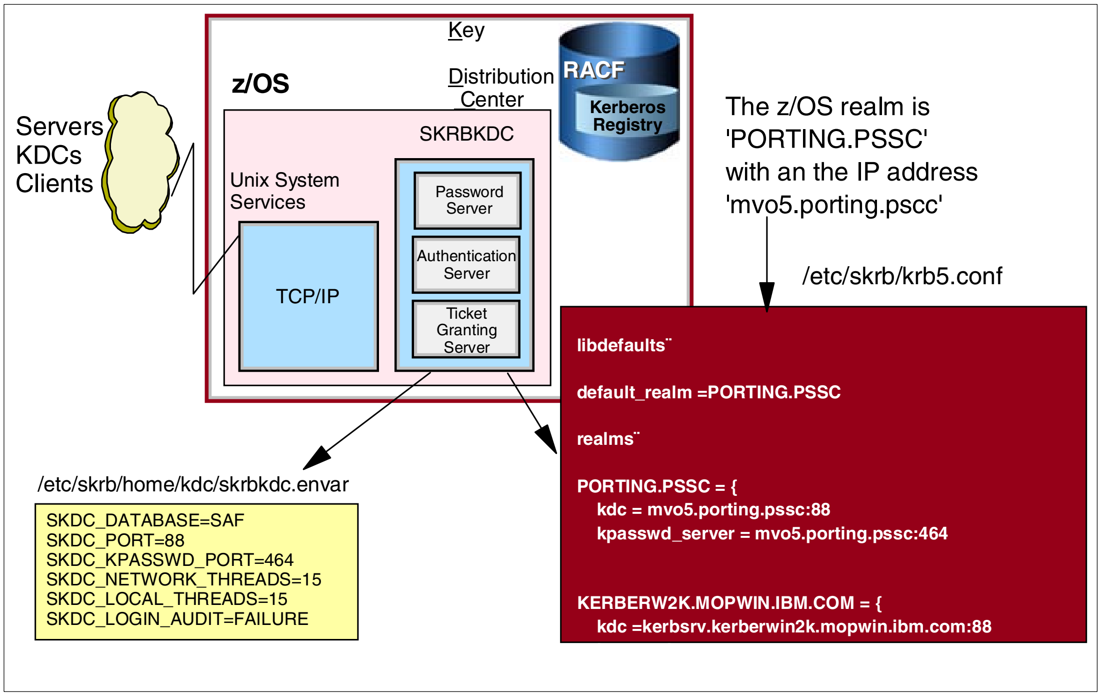
- Kerberos overview
- Strong cryptographic support
- XMEM credential support
- New and changed Kerberos APIs
- New and changed GSS-APIs
- Installation, configuration and administration changes
- New security server commands
- New status codes and messages
- New security server features
- SKRBKDC automatic start or restart
- KMIGRATE - UNIX utility

16.1 Kerberos overview

The z/OS SecureWay Server Network Authentication and Privacy Service is based on Kerberos Version 5.

The Kerberos authentication system was developed at MIT in the 80s. It is a trusted third-party authentication system whose main purpose is to allow clients and processes (principals) to prove their identity across an network without a interchange of secret information. All entities in Kerberos must be registered with an authentication server; in other words, each must have a secret key (password), which is shared only with an authentication server, here named “AS.” In order to obtain services from an application server, first of all, a client (principal) must acquire a *ticket* from the AS. So, the client presents this ticket to the application server, which can now perform the verification or validation of the client’s identity; in other words, it can authenticate the client. In more detail, the authentication process happens as shown in Figure 16-1 and can be described as follows:

1. A client sends to the AS a request for *credentials* for a given application server.
2. The AS returns the credentials (a ticket for the application server and a secret session key, here named “k1”), all encrypted by the client’s secret key.
3. The client transmits the ticket (information about the client plus k1, all encrypted by the application server’s secret key) plus an authenticator (a short-lived packet of information, encrypted by k1).
4. The application server decrypts the ticket, obtains k1, then decrypts the authenticator and verifies the client’s identity.
5. The process of authentication is complete. Optionally, k1 may also be used to accomplish confidentiality between the two parties.



```
SKDC_DATABASE=SAF
SKDC_PORT=88
SKDC_KPASSWD_PORT=464
SKDC_NETWORK_THREADS=15
SKDC_LOCAL_THREADS=15
SKDC_LOGIN_AUDIT=FAILURE
```

```
libdefaults¨

default_realm =PORTING.PSSC

realms¨

PORTING.PSSC = {
   kdc = mvo5.porting.pssc:88
   kpasswd_server = mvo5.porting.pssc:464


KERBERW2K.MOPWIN.IBM.COM = {
   kdc =kerbsrv.kerberwin2k.mopwin.ibm.com:88
```

Figure 16-1 z/OS Kerberos server setup

The procedure for obtaining, storing, and using tickets is divided into two steps:

1. When the client first logs in and enters his password, the client software uses the password to obtain a special ticket know as a *Ticket-Granting Ticke*t (TGT) from the Authentication Server.
2. When a client requires access to a service, the client software presents the TGT to the *Ticket-Granting Server* (TGS), which then issues a ticket for that particular service.

The Authentication Server (AS) and the Ticket-Granting Server (TGS) together form the *Key Distribution Center* (KDC), which maintains a database of principals and their secret keys. Since the KDC maintains a database of passwords (encryption secret keys) for users at the site, it is extremely important that it be installed on a carefully protected and physically secure machine.

Kerberos was primarily designed for *intra-enterprise* use. In this way, a client in one organization can be authenticated to a server in another. So, multiple authentication servers will exist, each responsible for a subset of users or servers in the system. The subset of the users and servers registered to a particular server is called a *realm*. The realm includes the KDC and all principals registered in the database. A realm name is normally a DNS domain or subdomain name, in upper-case. Cross-realm authentication allows a principal to prove its identity to a server registered in a different realm. In order to accomplish that, a *inter-realm key* must be established. A client is then able to obtain a TGT for the remote TGS from its own realm. For more information, see *z/OS SecureWay Security Server Network Authentication Service Administration*, SC24-5926.

16.2 Strong cryptography support

The Kerberos's authentication process is based on symmetric cryptography. It uses the Data Encryption Standard (DES) and its variant Triple DES (DES3) as its methods of encryption.

The DES was developed by IBM around 1974 and adopted as a national standard in 1977. It is designed to encipher and decipher blocks of data of 64 bits under control of a 64-bit key. DES3 is a minor variation of this standard. Since it is based on the DES algorithm, it is very easy to modify existing software to use DES3. It also has the advantage of proven reliability and a longer key length. DES3 is simply another mode of DES operation. The procedure for encryption is exactly the same as regular DES, but it is repeated three times. The data is encrypted with the first key, decrypted with the second key, and finally encrypted again with the third key. This is illustrated in Figure 16-2 on page 266.

Although the input key for DES is 64 bits long, the actual key used by DES is only 56 bits in length. The least significant (right-most) bit in each byte is a parity bit. These parity bits are ignored, so only the seven most significant bits of each byte are used, resulting in a key length of 56 bits. This means that the effective key strength for DES3 is actually 168 bits because each of the three keys contains 8 parity bits that are not used during the encryption process.

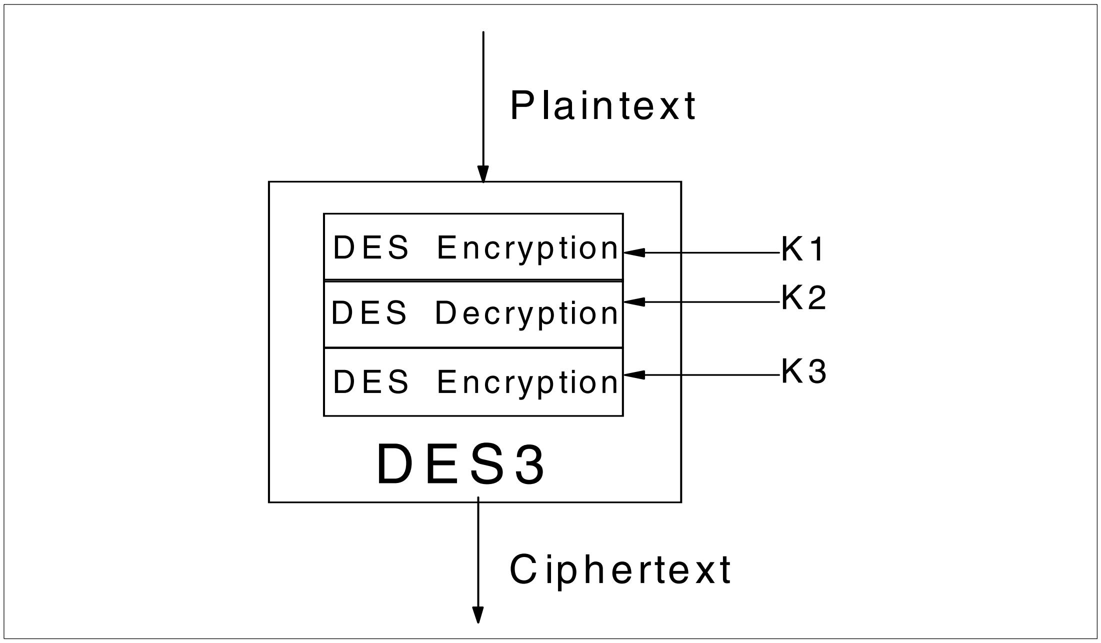


Figure 16-2 DES3 encryption process

In Network Authentication Service for z/OS Version 2 Release 1, 56-bit DES encryption is always supported. 168-bit DES3 encryption is always available for authentication, but may not be available for user data encryption because of US government export regulations. This means that session keys tickets may be in 56-bit DES.

The KDC attempts to use the same encryption type for the service ticket as was used for the TGT, in order to enable cross-realm support with realms that do not have DES3. All systems in the realm must support DES3 for tickets before DES3 ticket support can be enabled. In the same way, all systems in the realm must support DES3 for user data encryption before DES3 encryption for user data can be enabled.

16.3 XMEM credential support

The credentials cache holds Kerberos credentials (tickets, session keys, and other identifying information) in a semi-permanent store. The Kerberos runtime reads credentials from the cache as they are needed and stores new credentials in the cache as they are obtained. This way, the application does not have to manage the credentials itself.

For z/OS Version1 Release 2, Kerberos supports a new type of credentials cache, XMEM. An XMEM credentials cache is maintained in a data space by the Kerberos security server. The credentials cache can be read from any system in the sysplex, but can be updated only from the system that created the credentials cache. The credentials cache does not persist when the Kerberos security server terminates. The Kerberos security server periodically deletes credentials caches that contain only expired credentials. The `MODIFY SKRBKDC, DISPLAY CREDS` command can be used to display the current contents of the credentials data space.

Now, the possible credentials caches are three: FILE, MEMORY, and XMEM. The default credentials cache is FILE.

16.4 New and changed Kerberos APIs

To make it possible for applications programmers to create Kerberos-enable applications, a set of application programming interfaces (APIs) are available. These APIs allow programmers to create interfaces to the z/OS SecureWay Security Server Network Authentication Service, rendering viable the creation of a *Kerberized* environment. For more details, see *z/OS SecureWay Security Server Network Authentication Service Programming*, SC24-5927.

A whole new group of APIs, the Kerberos administration APIs, is introduced for z/OS Version1 Release 2. Additions to new groups and modifications to existing APIs are made. The following sections cover theses APIs, grouped as follows:

- The new APIs
- The changed APIs
- The Kerberos administration APIs

16.4.1 The new APIs

This section introduces the new Kerberos APIs in alphabetical order, and describes the purpose of each one. For more details, see *z/OS SecureWay Security Server Network Authentication Service Programming*, SC24-5927*.*

krb5_c_block_size	Returns the cipher block size
krb5_c_checksum_length	Returns the checksum length
krb5_c_decrypt	Decrypts a data block
krb5_c_encrypt	Encrypts a data block
krb5_c_encrypt_length	Returns the encrypted data length
krb5_c_enctype_compare	Compares two encryption types to determine if they are similar
krb5_c_keyed_checksum_types	Returns a list of keyed checksum types compatible with an encryption type
krb5_c_make_checksum	Generates the checksum for a data block
krb5_c_make_random_key	Generates a random encryption key
krb5_c_random_make_octets	Generates a random binary string
krb5_c_random_seed	Sets a new seed for the random number generator
krb5_c_string_to_key	Generates an encryption key from a text string
krb5_c_verify_checksum	Verifies the checksum for a data block
krb5_cc_set_default_name	Sets the default credentials cache name for the Kerberos context
krb5_dll_load	Loads the Kerberos runtime library
krb5_dll_unload	Unloads the Kerberos runtime library
krb5_free_checksum_contents	Releases the storage assigned to the contents of a checksum
krb5_free_cksumtypes	Releases the storage assigned to the contents of a checksum

krb5_free_data_contents	Releases the storage assigned to the contents of a Kerberos data object
krb5_free_enctypes	Releases the storage assigned to an array of encryption types
krb5_rd_req_verify	Processes a Kerberos AP_REQ message and verifies the application data checksum
krb5_read_password	Reads a password from the terminal in non-display mode

16.4.2 The changed APIs

This section introduces, in alphabetical order, all modified APIs, and defines the changes made to each one.

krb5_auth_con_set_req_cksumtype

Specifies two new input parameters, cksumtype, as follows:

CKSUMTYPE_NIST_SHA	NIST SHA checksum
CKSUMTYPE_HMAC_SHA1_DES3	DES3 HMAC checksum

krb5_auth_con_set_safe_cksumtype

New input parameters, cksumtype, are specified as follows:

CKSUMTYPE_NULL	Select the default checksum algorithm based upon the encryption key stored in the authentication context
CKSUMTYPE_HMAC_SHA1_DES3	DES3 HMAC checksum

The **krb5_mk_safe()** function requires a keyed checksum. In addition, the checksum must be compatible with the encryption key in the authentication context.

krb5_cc_default_name

The **krb5_cc_default_name()** routine returns the name of the default credentials cache for the Kerberos context. The default credentials cache is determined as follows:

- The name set by the **krb5_cc_set_default_name()** routine.
- The value of the KRB5CCNAME environment variable.
- The contents of the file specified by the _EUV_SEC_KRB5CCNAME_FILE environment variable (the file name defaults to $HOME/krb5ccname if _EUV_SEC_KRB5CCNAME_FILE is not set).
- A new credentials cache name is generated if no default name is found.

The function return value is NULL if an error occurred. Otherwise, it is the address of the default credentials cache name. This is a pointer to read-only storage and must not be freed by the application.

The **krb5_cc_default_name()** and **krb5_cc_set_default_name()** routines use storage within the Kerberos context to hold the default credentials cache name. Thus, these routines are not thread-safe unless a separate Kerberos context is used for each thread.

krb5_cc_resolve

The Kerberos runtime supports three credentials cache types: FILE, MEMORY, and XMEM.

krb5_cc_retrieve_cred

A new input parameter, flag, is specified as follows:

KRB5_TC_SUPPORTED_KTYPES

The encryption key type in the cache entry must be one of the encryption types specified by the **default_tgs_enctypes** value in the Kerberos configuration profile. If the **default_tgs_enctypes** value contains multiple encryption types, the list is processed from left to right and the first matching credential is returned.

krb5_free_ap_rep_enc_part

A new format is specified, as follows:

```
#include <skrb/krb5.h>
void krb5_free_ap_rep_enc_part (
krb5_context context,
krb5_ap_rep_enc_part *enc_part)
```

krb5_get_cred_via_tkt

A new format is specified, as follows:

```
#include <skrb/krb5.h>
krb5_error_code krb5_get_cred_via_tkt (
krb5_context context,
krb5_creds *tkt,
const krb5_flags kdc_options,
krb5_address *const *address
krb5_creds *in_cred,
krb5_creds **out_cred)
```

krb5_get_default_in_tkt_ktypes

The output parameter **ktypes** is specified as follows:

Returns an array of encryption types. The last entry in the array is ENCTYPE_NULL. The caller is responsible for freeing the array returned for this parameter, when it is no longer needed, by calling the **krb5_free_enctypes()** routine.

krb5_get_default_tgs_ktypes

The output parameter **ktypes** is specified as follows:

Returns an array of encryption types. The last entry in the array is ENCTYPE_NULL. The caller is responsible for freeing the array returned for this parameter, when it is no longer needed, by calling the **krb5_free_enctypes()** routine.

krb5_get_in_tkt_system

New input parameters, **enctypes**, are specified as follows:

ENCTYPE_DES_CBC_MD4	MD4 checksum with DES encryption
ENCTYPE_DES_HMAC_SHA1	SHA1 checksum with DES encryption and key derivation
ENCTYPE_DES3_CBC_SHA1	SHA1 checksum with DES3 encryption and key derivation

As a general rule, the application should not specify the encryption types. This allows the encryption type to be determined by the Kerberos configuration profile.

krb5_get_in_tkt_with_keytab

New input parameters, **enctypes** and **pre_auth_types**, are specified as follows:

ENCTYPE_DES_CBC_MD4	MD4 checksum with DES encryption
ENCTYPE_DES_HMAC_SHA1	SHA1 checksum with DES encryption and key derivation
ENCTYPE_DES3_CBC_SHA1	SHA1 checksum with DES3 encryption and key derivation

pre_auth_types

No preauthentication is done unless required by KDC policy (in which case the KDC provides the preauthentication types).

As a general rule, the application should not specify encryption or preauthentication types. This allows the encryption type to be determined by the Kerberos configuration profile and the preauthentication type to be determined by the KDC policy.

The first encryption type specified (either explicitly or through the Kerberos configuration profile) is used for preauthentication types that require an encryption key. If the KDC returns a list of encryption types, the first supported encryption type is used for preauthentication data.

krb5_get_in_tkt_with_password

New input parameters, **enctypes** and **pre_auth_types**, are specified as follows:

ENCTYPE_DES_CBC_MD4	MD4 checksum with DES encryption
ENCTYPE_DES_HMAC_SHA1	SHA1 checksum with DES encryption and key derivation
ENCTYPE_DES3_CBC_SHA1	SHA1 checksum with DES3 encryption and key derivation

pre_auth_types

No preauthentication is done unless required by KDC policy (in which case the KDC provides the preauthentication types).

As a general rule, the application should not specify encryption or preauthentication types. This allows the encryption type to be determined by the Kerberos configuration profile and the preauthentication type to be determined by the KDC policy.

The first encryption type specified (either explicitly or through the Kerberos configuration profile) is used for preauthentication types that require an encryption key. If the KDC returns a list of encryption types, the first supported encryption type is used for preauthentication data.

krb5_get_in_tkt_with_skey

New input parameters, **enctypes**, **pre_auth_types**, and **key**, are specified as follows:

ENCTYPE_DES_CBC_MD4	MD4 checksum with DES encryption
ENCTYPE_DES_HMAC_SHA1	SHA1 checksum with DES encryption and key derivation
ENCTYPE_DES3_CBC_SHA1	SHA1 checksum with DES3 encryption and key derivation

pre_auth_types

No preauthentication is done unless required by KDC policy (in which case the KDC provides the preauthentication types).

key

Specifies the key to be used. The default key table is used if NULL is specified for this parameter. The key must be the current encryption key for the client principal.

As a general rule, the application should not specify encryption or preauthentication types. This allows the encryption type to be determined by the Kerberos configuration profile and the preauthentication type to be determined by the KDC policy.

The first encryption type specified (either explicitly or through the Kerberos configuration profile) is used for preauthentication types that require an encryption key. If the KDC returns a list of encryption types, the first supported encryption type is used for preauthentication data.

krb5_kt_add_entry

A new input parameter, **entry**, is specified as follows:

entry

Specifies the entry to be added to the key table. The application is responsible for setting the *principal*, *vn*o, and *key* fields in the entry. The **krb5_kt_add_entry()** routine sets the *timestamp* field to the current time.

krb5_mk_priv

The encryption key is obtained from the local subkey, the remote subkey, or the session key, in that order. The application is responsible for setting a checksum type in the authentication context that is compatible with the encryption key. For example, an error is returned if a DES3 encryption key is used with a DES checksum type.

krb5_mk_req

A new input parameter, **ap_req_options**, is specified as follows:

ap_req_options

AP_OPTS_USE_SESSION_KEY - Use session key instead of server key for the service ticket. The credentials must include a ticket that is encrypted in the session key.

The **krb5_mk_req()** routine generates an AP_REQ message. The checksum of the application data is included in the authenticator that is part of the AP_REQ message. This message is then sent to the partner application, which calls the **krb5_rd_req()** routine to validate the authenticity of the message. The checksum method set in the authentication context is used to generate the checksum.

krb5_mk_req_extended

New input parameters, **ap_req_options** and **appl_data**, are specified as follows:

ap_req_options

AP_OPTS_USE_SESSION_KEY - Use session key instead of server key for the service ticket. The credentials must include a ticket that is encrypted in the session key.

appl_data

Specifies the application data whose checksum is to be included in the authenticator. Specify NULL for this parameter if no checksum is to be included in the authenticator.

The **krb5_mk_req()** routine generates an AP_REQ message. The checksum of the application data is included in the authenticator that is part of the AP_REQ message. This message is then sent to the partner application, which calls the **krb5_rd_req()** routine to validate the authenticity of the message. The checksum method set in the authentication context is used to generate the checksum.

krb5_mk_safe

The encryption key is obtained from the local subkey, the remote subkey, or the session key, in that order. The application is responsible for setting a checksum type in the authentication context that is compatible with the encryption key. For example, an error is returned if a DES3 encryption key is used with a DES checksum type.

krb5_rc_resolve

After successfully calling **krb5_rc_resolve()**, the application should call either the **krb5_rc_recover()** or the **krb5_rc_initialize()** routine. This initializes the in-storage replay cache structures. The use of in-storage structures significantly improves performance, but means that multiple replay cache handles should not be opened for the same replay cache.

krb5_rd_priv

A new output **out_data** parameter is specified as follows:

out_data

Returns the application data supplied to the **krb5_mk_priv()** routine. The application should release the data when it is no longer needed by calling the **krb5_free_data_contents()** routine.

krb5_rd_safe

A new output parameter, **out_data**, is specified as follows:

out_data

Returns the application data supplied to the **krb5_mk_priv()** routine. The application should release the data when it is no longer needed by calling the **krb5_free_data_contents()** routine.

krb5_set_default_in_tkt_ktypes

New input parameters are specified as follows:

ENCTYPE_DES_CBC_MD4	DES encryption with an MD4 checksum
ENCTYPE_DES_HMAC_SHA1	DES encryption with SHA1 checksum
ENCTYPE_DES3_CBC_SHA1	DES3 encryption with SHA1 checksum

The **krb5_set_default_in_tkt_ktypes()** routine sets the default encryption types used when requesting the initial ticket from the KDC. In order to interoperate with older Kerberos V5 servers, you should include ENCTYPE_DES_CBC_CRC as one of the encryption types.

krb5_set_default_realm

A new format is specified, as follows:

```
#include <skrb/krb5.h>
krb5_error_code krb5_set_default_realm (
krb5_context context,
const char *realm)
```

krb5_set_default_tgs_ktypes

New input parameters are specified, as follows:

ENCTYPE_DES_CBC_MD4	DES encryption with an MD4 checksum
ENCTYPE_DES_HMAC_SHA1	DES encryption with SHA1 checksum
ENCTYPE_DES3_CBC_SHA1	DES3 encryption with SHA1 checksum

The **krb5_set_default_in_tkt_ktypes()** routine sets the default encryption types used when requesting the initial ticket from the KDC. In order to interoperate with older Kerberos V5 servers, you should include ENCTYPE_DES_CBC_CRC as one of the encryption types.

krb5_sname_to_principal

A new format is specified, as follows:

```
#include <skrb/krb5.h>
krb5_error_code krb5_sname_to_principal (
krb5_context context,
const char *hostname,
const char *sname,
krb5_int32 type,
krb5_principal *ret_princ)
```

krb5_us_timeofday

The function return value is zero if no errors occurred. Otherwise, it is a Kerberos error code.

16.4.3 The Kerberos administration APIs group

In this section, an entirely new group of APIs, the Kerberos administration APIs, is introduced. The APIs are presented in alphabetical order, and the purpose of each is described. For more details, see *z/OS SecureWay Security Server Network Authentication Service Programming*, SC24-5927.

kadm5_chpass_principal	Changes the password for a principal entry in the Kerberos database
kadm5_create_policy	Creates a policy entry in the Kerberos database
kadm5_create_principal	Creates a principal entry in the Kerberos database
kadm5_delete_policy	Deletes a policy entry from the Kerberos database
kadm5_delete_principal	Deletes a principal entry from the Kerberos database
kadm5_destroy	Closes a session with the Kerberos administration server
kadm5_free_key_list	Frees a list of keys
kadm5_free_name_list	Frees a list of names
kadm5_free_policy _ent	Releases storage allocated for a policy entry
kadm5_free_principal_ent	Releases storage allocated for a principal entry
kadm5_get_policies	Returns a list of policies matching the specified search expression
kadm5_get_policy	Returns information from a policy entry in the Kerberos database
kadm5_get_principal	Returns information from a principal entry in the Kerberos database
kadm5_get_principals	Returns a list of principals matching the specified search expression
kadm5_init_with_creds	Establishes a session with the Kerberos administration server using a credentials cache for authentication
kadm5_init_with_password	Establishes a session with the Kerberos administration server using a password for authentication
kadm5_init_with_skey	Establishes a session with the Kerberos administration server using a key table for authentication
kadm5_modify_policy	Modifies a policy entry in the Kerberos database
kadm5_modify_principal	Modifies a principal entry in the Kerberos database

kadm5_randkey_principal	Generates a new set of random keys for a principal
kadm5_rename_principa	Renames a principal entry in the Kerberos database
kadm5_setkey_principal	Sets the key for a principal entry in the Kerberos database

16.5 New and changed GSS-APIs

The generic security service application programming interface (GSS-API) provides security services to applications using peer-to-peer communications. Using GSS-API routines, applications can perform these operations:

- Enable an application to determine another application's user identification
- Enable an application to delegate access rights to another application
- Apply security services, such as confidentiality and integrity, on a per-message basis

A secure connection between two peers (two communicating applications) is represented by a data structure called a *security context*. The operational paradigm in which GSS-API operates is as follows:

> A typical GSS-API caller is itself a communication protocol, calling on GSS-API to protect its communications with authentication, integrity, and/or confidentiality security services. A GSS-API caller accepts tokens provided to it by its local GSS-API implementation and transfers the tokens to a peer on a remote system; that peer passes the received tokens to its local GSS-API implementation for processing. The security services available through GSS-API in this fashion are implementable (and have been implemented) over a range of underlying mechanisms based on secret-key and public-key cryptographic technologies.

There are two types of GSS-API routines:

- Standard GSS-API routines, which are defined in Internet RFC 2078, *Generic Security Service Application Program Interface, Version 2* and Internet RFC 1509, *Generic Security Service API: C binding*s. These routines have the prefix **gss_**.
- Kerberos extensions to the GSS-API. These are additional routines that enable an application to use Kerberos security services. These routines have the prefix **gss_krb5_**.

For z/OS Version 1 Release 2, additions and modifications to existing APIs are made. For more details see *z/OS SecureWay Security Server Network Authentication Service Programming*, SC24-5927.

In the following sections we discuss these topics:

- The new **gss_** APIs
- The changed **gss_** APIs
- The new **gss_krb5_** APIs

16.5.1 The new gss_ APIs

This section introduces the new **gss_** APIs in alphabetical order, and describes the purpose of each one. For more details, see *z/OS SecureWay Security Server Network Authentication Service Programming*, SC24-5927.

gss_export_cred	Creates a credential token for a GSS-API credential
gss_export_name	Exports a mechanism name as an opaque token

gss_export_sec_context Creates a security context token for a GSS-API security context

gss_import_cred Creates a GSS-API credential from a credential token created by the **gss_export_cred()** routine

gss_import_sec_context Creates a GSS-API security context from a security context token created by the **gss_export_sec_context()** routine

16.5.2 The changed gss_ APIs

This section introduces, in alphabetical order, all modified **gss_** APIs, and defines the modifications made to each one.

gss_accept_sec_context

New output **src_name** and **ret_flags** parameters are specified, as follows:

src_name

The application should release the name when it is no longer needed by calling the **gss_release_name()** routine.

ret_flags

GSS_C_TRANS_FLAG - If this flag is set, the **gss_export_sec_context()** function can be used to export the security context. The **gss_export_sec_context()** function is not available if this flag is not set.

gss_context_time

The Kerberos security mechanism supports context expiration and returns the time remaining before the underlying service ticket expires, if the context was created by **gss_accept_sec_context()**, or the lesser of the requested expiration time and the ticket expiration time, if the context was created by **gss_init_sec_context()**.

gss_display_name

New output parameters, **output_name_buffer** and **output_name_type**, are specified as follows:

output_name_buffer

Return buffer for the character string. The **gss_release_buffer()** routine should be called to release the storage when it is no longer needed.

output_name_type

Returns the name type corresponding to the returned character string. The *gss_OID* value returned for this parameter points to read-only storage and must not be released by the application. Specify NULL if the name type is not needed.

gss_get_mic

A new input parameter, **qop_req**, is specified as follows:

qop_req

GSS_C_QOP_DEFAULT should always be specified unless it is necessary to select a specific quality-of-protection algorithm, in which case the application must ensure that the selected algorithm is compatible with the security mechanism associated with the security context.

The Kerberos security mechanism supports four integrity algorithms. The default algorithm can be requested by specifying GSS_C_QOP_DEFAULT, which is equivalent to specifying GSS_KRB5_INTEG_C_QOP_DEFAULT.

The integrity algorithms are as follows:

GSS_KRB5_INTEG_C_QOP_DEFAULT
Default integrity algorithm (QOP_DES_MD5 for a DES session key or QOP_HMAC_SHA1 for a DES3 session key)

GSS_KRB5_INTEG_C_QOP_HMAC_SHA1
DES3_HMAC of SHA1 checksum

The encryption key associated with the security context determines which quality-of-protection algorithms are available. The GSS_KRB5_INTEG_C_QOP_HMAC_SHA1 algorithm requires a 168-bit DES3 key.

gss_import_name

A new **input_name_type** input parameter is specified as follows:

input_name_type

GSS_C_NT_EXPORT_NAME - specifies an exported name created by the **gss_export_name()** routine.

gss_init_sec_context

New output parameters, **actual_mech_type** and **ret_flags**, are specified as follows:

actual_mech_type

Returns the security mechanism to be used with the context. The **gss_OID** value returned for this parameter points to read-only storage and must not be released by the application. Specify NULL for this parameter if the actual mechanism type is not needed.

ret_flags

GSS_C_TRANS_FLAG - If this flag is set, the **gss_export_sec_context()** function can be used to export the security context. The **gss_export_sec_context()** function is not available if this flag is not set.

gss_inquire_cred_by_mech

A new status code is specified as follows:

GSS_S_CREDENTIALS_EXPIRED

The credentials have expired. Credential information is still returned for an expired credential but the lifetime value is returned as zero.

gss_unwrap

A new output parameter, **qop_state**, is specified as follows:

qop_state

Returns the quality of protection applied to the message. Specify NULL for this parameter if the quality of protection is not needed.

gss_verify_mic

A new output parameter, **qop_state**, is specified as follows:

qop_state

Returns the quality of protection applied to the message. Specify NULL for this parameter if the quality of protection is not needed.

gss_wrap

A new input parameter, **qop_req**, is specified as follows:

qop_req

GSS_C_QOP_DEFAULT should always be specified unless it is necessary to select a specific quality-of-protection algorithm, in which case the application must ensure that the selected algorithm is compatible with the security mechanism associated with the security context.

The Kerberos security mechanism supports four integrity algorithms and two sealing algorithms. The quality of protection value is formed by or'ing together one of the integrity algorithm values and one of the sealing algorithm values The default algorithms can be requested by specifying GSS_C_QOP_DEFAULT, which is equivalent to specifying GSS_KRB5_INTEG_C_QOP_DEFAULT or GSS_KRB5_CONF_C_QOP_DEFAULT.

The integrity algorithms are:

GSS_KRB5_INTEG_C_QOP_DEFAULT
Default integrity algorithm (QOP_DES_MD5 for a DES session key or QOP_HMAC_SHA1 for a DES3 session key)

GSS_KRB5_INTEG_C_QOP_HMAC_SHA1
DES3_HMAC of SHA1 checksum

The sealing algorithms are:

GSS_KRB5_CONF_C_QOP_DEFAULT
Default sealing algorithm (QOP_DES for a DES session key or QOP_DES3_KD for a DES3 session key)

GSS_KRB5_CONF_C_QOP_DES
DES encryption

GSS_KRB5_CONF_C_QOP_DES3_KD
DES3 encryption with key derivation

The encryption key associated with the security context determines which quality-of-protection algorithms are available. The GSS_KRB5_CONF_C_QOP_DES algorithms require a 56-bit DES key. The GSS_KRB5_INTEG_C_QOP_HMAC_SHA1, and GSS_KRB5_CONF_C_QOP_DES3_KD algorithms require a 168-bit DES3 key.

gss_wrap_size_limit

A new input parameter, **qop_req**, is specified as follows:

qop_req

GSS_C_QOP_DEFAULT should always be specified unless it is necessary to select a specific quality-of-protection algorithm, in which case the application must ensure that the selected algorithm is compatible with the security mechanism associated with the security context.

The Kerberos security mechanism supports the following changes to the integrity algorithms and sealing algorithms. The quality of protection value is formed by or'ing together one of the integrity algorithm values and one of the sealing algorithm values The default algorithms can be requested by specifying GSS_C_QOP_DEFAULT, which is equivalent to specifying GSS_KRB5_INTEG_C_QOP_DEFAULT or GSS_KRB5_CONF_C_QOP_DEFAULT.

The integrity algorithms are:

GSS_KRB5_INTEG_C_QOP_DEFAULT
Default integrity algorithm (QOP_DES_MD5 for a DES session key or QOP_HMAC_SHA1 for a DES3 session key)

GSS_KRB5_INTEG_C_QOP_HMAC_SHA1
DES3_HMAC of SHA1 checksum

The sealing algorithms are:

GSS_KRB5_CONF_C_QOP_DEFAULT
Default sealing algorithm (QOP_DES for a DES session key or QOP_DES3_KD for a DES3 session key)

GSS_KRB5_CONF_C_QOP_DES
DES encryption

GSS_KRB5_CONF_C_QOP_DES3_KD
DES3 encryption with key derivation

The encryption key associated with the security context determines which quality-of-protection algorithms are available. The GSS_KRB5_CONF_C_QOP_DES algorithms require a 56-bit DES key. The GSS_KRB5_INTEG_C_QOP_HMAC_SHA1 and GSS_KRB5_CONF_C_QOP_DES3_KD algorithms require a 168-bit DES3 key.

16.5.3 The new gss_krb5_ APIs

This section introduces the new **gss_ krb5_** APIs in alphabetical order, and describes the purpose of each one. For more details, see *z/OS SecureWay Security Server Network Authentication Service Programming*, SC24-5927*.*

gss_krb5_acquire_cred_ccache Acquires a GSS-API credential using a Kerberos credentials cache.

gss_krb5_ccache_name Sets the default credentials cache name for use by the Kerberos mechanism.

gss_krb5_copy_ccache Copies the tickets from the Kerberos credentials cache associated with a GSS-API credential.

16.6 Installation, configuration, and administration changes

In this section we discuss the changes to the installation, configuration, and administration processes for Network Authentication Service for z/OS.

16.6.1 Installation changes

After you have installed z/OS, use the following steps to make Network Authentication Service operational:

1. Copy the SKRBKDC sample configuration file from /usr/lpp/skrb/examples/krb5.conf to the configuration file pointed to by the environment variable KRB5_CONFIG (or to /etc/skrb/krb5.conf if the KRB5_CONFIG environment variable is not specified).

   The file permissions should allow everyone to read the file and only the administrator to update it.

2. Copy the SKRBKDC environment variable definitions from usr/lpp/skrb/examples/skrbkdc.envar to /etc/skrb/home/kdc/envar.

   The file permissions should allow only the administrator to read and update the file. Set the TZ and RESOLVER_CONFIG values for your installation.

3. Before starting the SKRBKDC started task, be sure that the realm definitions and other configuration and RACF items are completed for your installation. Refer to the appropriate sections of *z/OS SecureWay Security Server RACF Security Administrator's Guide*, SA22-7683 and supporting publications for details about updating the RACF database template and the Dynamic Parsing task before using Network Authentication Service for z/OS.
4. If your installation uses the SERVAUTH RACF class profiles to control access to TCP/IP ports and stacks, also use the RACF publications as a guide to update the TCP/IP resource permissions needed for Network Authentication Service users and KDCs.
5. With RACF as the external security manager, the IRR.RUSERMAP resource in the FACILITY class must be defined. The test_server and test_delegate system IDs (for example, KRBSRV4 and KRBDLG4) must have RACF READ access to IRR.RUSERMAP resource to use the KRB5_SERVER_KEYTAB variable set to 1. To define IRR.RUSERMAP and grant READ authority to all system users, issue the following commands:

   ```
   REDEFINE FACILITY IRR.RUSERMAP UACC(READ)
   SETROPTS RACLIST(FACILITY)REFRESH
   ```

   See "Security runtime environment variables" on page 284 for more on the KRB5_SERVER_KEYTAB environment variable.
6. Define the RACF Remote Sharing Facility (RRSF) for the local system, even if you do not plan to set up an RRSF network. An RRSF local node must be defined in order to generate the corresponding Kerberos secret key whenever users change their password. Refer to *z/OS SecureWay Security Server RACF Security Administrator's Guide*, SA22-7683 for information on defining the local RRSF node.
   a. Create the SKRBKDC user ID. This user ID must have UID(0).

   ```
   ADDUSER SKRBKDC DFLTGRP(SYS1) NOPASSWORD OMVS(UID(0)PROGRAM('/bin/sh')
   HOME('/etc/skrb/home/kdc'))
   ```

   b. Activate the APPL class if it is not already active.

   ```
   SETROPTS CLASSACT(APPL)RACLIST(APPL)
   ```
7. Define the SKRBKDC application and set the universal access to READ. Alternately, you can set the universal access to NONE and define individual groups or users to the SKRBKDC application. Users must have access to the SKRBKDC application to use the **kpasswd** Kerberos command to change their passwords.

   ```
   RDEFINE APPL SKRBKDC UACC(READ)
   ```
8. Activate the PTKTDATA class if it is not already active.

   ```
   SETROPTS CLASSACT(PTKTDATA) RACLIST(PTKTDATA)
   ```
9. Define the PassTicket data for the SKRBKDC application. PassTickets are used internally by the Kerberos security server when the user password is changed. A PassTicket is never given to a user by the Kerberos security server. The secured signon key can be any valid DES key, as described in *z/OS SecureWay Security Server RACF Security Administrator's Guide*, SA22-7683.

   ```
   RDEFINE PTKTDATA SKRBKDC UACC(NONE) SSIGNON(KEYMASKED(3734343237343131))
   ```
10. Refresh the APPL and PTKTDATA classes.

   ```
   SETROPTS RACLIST(APPL PTKTDATA)REFRESH
   ```
11. Define the SKRBKDC started task and associate it with the SKRBKDC user ID. Define the SKRBWTR started task and associate it with the SKRBKDC user ID if you plan to perform component tracing with the SKRBWTR procedure.

   ```
   RDEFINE STARTED SKRBKDC.**STDATA(USER(SKRBKDC))
   ```

```
RDEFINE STARTED SKRBWTR.**STDATA(USER(SKRBKDC))
```

12. Refresh the STARTED Class.

    ```
    SETROPTS RACLIST(STARTED)REFRESH
    ```

    Because the KDC function in the DCE Security Server uses port 88, and the KDC function in the Network Authentication Service defaults to port 88, you must take the following action to avoid a conflict.

    – If you are not running a DCE Security Server on your system, you will not have a conflict. Skip this list.

    – If you are running a DCE Security Server on your system, but are not using the KDC portion of it, add the line `SECD_ENABLE_KDC=0` to the DCE Security Server envar file to disable usage of port 88. DCE, however, generally uses Remote Procedure Call (RPC) for security operations and does *not* use the DCE KDC function.

    – If you are running a DCE Security Server on your system, and you *are* using the KDC portion of it, change port 88 in the Network Authentication Service KDC envar file in /etc/skrb/home/kdc to comply with the operation of the network at your installation. All clients using the DCE Security Server must do the same.

    Because DCE uses the same commands and environment variables as Network Authentication Service, you must perform these steps:

    a. Determine which user IDs will be using Network Authentication Service.

    b. Customize the LOGON procedures for the Network Authentication Service users:

       - Remove the DCE REXX exec data set if it is present in the SYSEXEC DD name concatenation.

       - Add the Network Authentication Service REXX exec data set, EUVF.SEUVFEXC, to the SYSEXEC DD name concatenation.

       - 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.

    c. Customize the Communications Server PROFILE DD name member to ensure that the selected ports for the KDC (usually Port 88) and the KPASSWD port (usually Port 464) are reserved for OMVS.

13. The following data sets need APF authorization:

    – EUVF.SEUVFLNK

    – CEE.SCEERUN

16.6.2 Configuration changes

In the configuration profile, there are new supported checksum and encryption types. See RFC 1510 for more information about the checksum algorithms.

Supported checksum types are:

- crc32
- rsa-md4
- rsa-md4-des
- descbc
- rsa-md5
- rsa-md5-des
- nist-sha
- hmac-sha1-des3

Supported encryption types are:

- des-cbc-crc
- des-cbc-md4
- des-cbc-md5
- des-hmac-sha1
- des3-cbc-sha1

z/OS SecureWay Security Server Network Authentication Service Administration, SC24-5926 has more details about the configuration profile.

Configuration profile file sections

Table 16-1 describes the usage of each section of the configuration profile file.

Table 16-1 Sections of the configuration profile file

Section	Usage
[libdefaults]	This section provides defaults for the Kerberos runtime routines.
[realms]	This section defines each of the realms that can be reached from the local realm. For each realm, one or more key distribution center (KDC) hosts must be defined. The [realms] section is used if no DNS or LDAP server is available or if the desired Kerberos service is not found using the DNS or LDAP server.
[domain_realm]	This section defines the mapping between DNS names and Kerberos realm names. The [domain_realm] section is used if no DNS or LDAP server is available or if the desired mapping is not found using the DNS or LDAP server.
[capath]	This section defines connection paths between realms. This section is not required if the Kerberos realms are arranged in a hierarchical configuration or if each realm has a peer connection to every other realm. Even in a hierarchical configuration, this section should be defined if there are direct connections between realms.

The following changes and additions have been made to parameters in the configuration profile file:

[libdefaults] section

- **kpasswd_use_tcp**

  Set this value to 1 to use TCP stream connections instead of UDP datagrams when sending a request to the password change server. If a TCP connection cannot be established with the server, the runtime tries again by sending a UDP datagram to the password change server. Set this value to 0 to always use UDP datagrams. The default is 1.

- **safe_checksum_type**

  Specifies the default checksum type for a safe request. The default is rsa-md5-des. The specified checksum type must be compatible with the session key encryption type if the checksum uses an encrypted hash. For example, the rsa-md5-des checksum type requires a DES session key (such as des-cbc-crc) while the hmac-sha1-des3 checksum type requires a DES3 session key (such as des3-cbc-sha1).

- **use_dns_lookup**

  Set this value to 1 to use the DNS name server to locate the KDC and to resolve host names. The KDC is located using SRV records, and host names are resolved to realm names using TXT records. The [realms] and [domain_realm] sections are used if the resolution is unsuccessful using the DNS name server. Set this value to 0 to bypass the DNS lookup step. The default is 0. The priority value for SRV records is used to order the service records. Entries with the same priority are randomly selected each time the client needs to contact a Kerberos server.

- **use_ldap_lookup**

  Set this value to 1 to use the LDAP directory to locate the KDC and to resolve host names. The [realms] and [domain_realm] sections are used if the resolution is unsuccessful. Set this value to 0 to bypass the LDAP lookup step. The default is 0. If both LDAP and DNS are used, LDAP is checked first, followed by DNS. The *ldap_server* value must also be specified to use LDAP lookup. LDAP directory entries are randomly selected each time the client needs to contact a Kerberos server.

[realms] section

- **realm**

  The *realm* value is a Kerberos realm name. The value is a group definition that defines the Kerberos servers for the realm. Each realm that can be contacted by applications on the local system must have an entry in the [realms] section of the configuration file unless DNS or LDAP lookup is enabled. The group entry consists of one or more occurrences of *kdc*, *admin_server*, and *kpasswd_server* names. Entries are randomly selected each time the client needs to contact a Kerberos server.

  The value for each *kdc* name entry is the host name, the port assigned to the KDC on that system, and the protocol (UDP or TCP), separated by colons. If the port is omitted, it defaults to 88. If the protocol is omitted, the entry can be used with both protocols.

  The value for each *admin_server* entry is the host name and the port assigned to the administration service on that system, separated by a colon. If the port is omitted, it defaults to 749. The protocol is always TCP for the administration service.

  The value for each *kpasswd_server* entry is the host name, the port assigned to the password service on that system, and the protocol (UDP or TCP), separated by colons. If the port is omitted, it defaults to 464. If the protocol is omitted, the entry can be used with both protocols.

z/OS SecureWay Security Server Network Authentication Service Administration, SC24-5926 has more details about the configuration profile.

Strong encryption

The Kerberos security server uses the strongest encryption algorithm available when encrypting a ticket. The SKDC_TKT_ENCTYPES environment variable for the SKRBKDC started task specifies the encryption types to be used for ticket-granting tickets and for service tickets. This is a list of one or more encryption types specified from most-preferred to least-preferred. When generating a ticket, the KDC selects the first entry in this list that has a key available for the server specified in the ticket. Thus, the use of 168-bit DES3 encryption can be controlled on an individual server basis when necessary. For example, if a foreign realm does not support 168-bit DES3 encryption, the *krbtgt/foreign-realm@local-realm* principal entry in the KDC registry database contains just a 56-bit DES key and not a 168-bit DES3 key.

The default_tkt_enctypes value in the Kerberos configuration profile specifies the encryption types to be used for session keys in initial ticket-granting tickets. This is a list of one or more encryption types specified from most-preferred to least-preferred. The KDC selects the first supported encryption type in the list when it generates the session key for an initial ticket-granting ticket. This encryption type is also used for preauthentication information.

The default_tgs_enctypes value in the Kerberos configuration profile specifies the encryption types to be used for session keys in service tickets. This is a list of one or more encryption types specified from most-preferred to least-preferred. The KDC selects the first supported encryption type in the list when it generates the session key for a service ticket.

To enable 168-bit DES3 support for tickets, specify the following in the SKRBKDC environment file (/etc/skrb/home/kdc/envar):

```
SKDC_TKT_ENCTYPES=des3-cbc-sha1,des-cbc-crc
```

To enable 168-bit DES3 support for authentication information, specify the following in the Kerberos configuration profile (/etc/skrb/krb5.conf):

```
default_tkt_enctypes =des3-cbc-sha1,des-cbc-crc
```

To enable 168-bit DES3 support for user data, specify the following in the Kerberos configuration profile (/etc/skrb/krb5.conf):

```
default_tgs_enctypes =des3-cbc-sha1,des-cbc-crc
```

To enable DES3 key generation, use the following RACF command:

```
SETROPTS KERBLVL(1)
```

Do not enable 168-bit DES3 ticket support until the Kerberos runtimes for all systems in the realm support DES3 encryption. Otherwise, you can obtain tickets that cannot be processed on a given system. In addition, do not enable DES3 encryption support for user data unless all systems in the realm support DES3 encryption for user data. Otherwise, you can obtain session keys that are unusable for exchanging encrypted data. This means that all systems sharing the database must be running z/OS Version 1 Release 2 or later.

When granting a service ticket, the KDC attempts to use the same encryption algorithm for the service ticket that was used for the ticket-granting ticket. This enables cross-realm encryption compatibility with realms that do not support the same encryption algorithms as the local realm. If the server principal does not have a key for that encryption type, then the KDC selects a key from the list specified by the SKDC_TKT_ENCTYPES environment variable.

GSS-API supports 56-bit DES and 168-bit DES3 encryption. To enable 168-bit DES3 encryption, update the default_tgs_enctypes value in the Kerberos configuration profile to specify both des3-cbc-sha1 and des-cbc-crc. The actual encryption type selected for a particular GSS-API security context is the intersection of the default_tgs_enctypes list and the available encryption keys for the target principal. For example, if the default_tgs_enctypes value is des3-cbc-sha1,des-cbc-crc, and the target principal has just a 56-bit DES key, the encryption type for the GSS-API security context is des-cbc-crc.

Security runtime configuration

The _kerberos service entries define KDC instances, while the _kerberos-adm service entries define administration service instances, and _kpasswd entries define the password change service instances. Since the z/OS implementation combines the KDC and the password change server, each instance of the Kerberos security server provides both the KDC and the password change services. At the present time, there is no administration service support on z/OS. The server entries are tried in priority order (0 is the highest priority). Server entries with the same priority are tried in a random order. _udp protocol records are required for the _kerberos and _kpasswd services, while _tcp protocol records are required for the _kerberos-adm service. _tcp protocol records should be present if the server supports TCP as well as UDP requests (the z/OS Kerberos security server supports both UDP and TCP requests).

Security server configuration

A step to verify that Network Authentication Service works was introduced in this item, after the start of the SKRBKDC started task.

Enter these commands from the OMVS environment:

- `kinit principal-name`
- `klist`

Unless these commands produce error messages, your setup is complete.

> **Note:** The Network Authentication Service administrator must create the Kerberos principal using the `ADDUSER` command and then assign an initial password using the `ALTUSER` command before the `kinit` command can be used.

The *z/OS SecureWay Security Server Network Authentication Service Administration*, SC24-5926 has more details about the configuration.

Security runtime environment variables

Environment variables can be defined in the current command shell using the `export` command. They can also be defined in an environment variable (envar) file, which is processed during security runtime initialization. Any variables defined through the shell override the same variables in the envar file. The _EUV_ENVAR_FILE environment variable can be used to specify the location of the envar file. By default, the $HOME/envar file is used.

z/OS generally sets environment variables in either /etc/profile (system-wide settings) or in $HOME/.profile (user-specific settings). Environment variables defined in either place override the same variables in the envar file.

Table 16-2 describes the security runtime environment variables that were changed or added.

Table 16-2 Security runtime environment variables

Environment variable	Explanation
_EUV_SVC_MSG_FACILITY	Specifies the facility class for messages written to the system logging facility. The valid facility classes are: KERN, USER, MAIL, NEWS, UUCP, DAEMON, AUTH, CRON, LPR, LOCAL0, LOCAL1, LOCAL2, LOCAL3, LOCAL4, LOCAL5, LOCAL6, LOCAL7. The default is USER.
_EUV_SVC_MSG_IDENTITY	Specifies the identity string prefixed to messages written to the system logging facility. The default is SKRB.

Environment variable	Explanation
_EUV_SVC_MSG_LOGGING	Specifies the target where messages are logged. The default is to write informational messages to stdout and error messages to stderr. The following values may be specified: NO_LOGGING = Suppress all messages STDOUT_LOGGING = Write all messages (informational and error) to stdout and also write error messages to stderr STDERR_LOGGING = Write informational messages to stdout and error messages to stderr. SYSTEM_LOGGING = Write all messages to the system logging facility (`syslogd` daemon).
KRB5_SERVER_KEYTAB	If this environment variable is set to 1, the **gss_accept_sec_context()** and **krb5_rd_req()** routines use a local instance of the Kerberos security server to decrypt service tickets instead of obtaining the key from a key table. The application must have at least READ access to the IRR.RUSERMAP facility in order to use this capability. **The Kerberos principal associated with the current system identity must be the same as the Kerberos principal in the service ticket. The key table is used if the Kerberos principal for the system identity is not the same as the Kerberos principal for the service ticket.**

Security server environment variables

The environment variables, shown in Table 16-3, are supported for the SKRBKDC started task. These variables are specified in /etc/skrb/home/kdc/envar.

Table 16-3 describes the security server environment variables that were changed or added.

Table 16-3 Security server environment variables

Environment variable	Explanation
SKDC_CREDS_SIZE	Specifies the credentials data space size in kilobytes, with a minimum value of 1024, a maximum value of 2097148, and a default value of 20480. The Kerberos security server stores cross-memory credentials in this data space.
SKDC_KADMIN_PORT	Specifies the administration service port number. If this environment variable is not defined, the administration service port is obtained from the *kerberos-adm* entry in the TCP/IP services files. If this entry is not defined, the administration service port defaults to 749.
SKDC_NETWORK_POLL	Specifies the network interface poll interval in minutes and defaults to 5. The security server queries the network configuration at the end of each poll interval to detect new network interfaces or the activation of a failed network interface.
SKDC_TKT_ENCTYPES	Specifies the encryption types to be used for ticket-granting tickets and for service tickets. This is a list of one or more encryption types separated by commas, specified from most-preferred to least-preferred. When generating a ticket, the KDC selects the first entry in the list that is available for the server specified in the ticket. The KDC uses des-cbc-crc if this environment variable is not defined.

LDAP schema definitions

Table 16-4 shows the LDAP object classes and Table 16-5 shows the LDAP attributes that were changed or added.

Table 16-4 LDAP object classes

Object	Requires	Allows
eSAP	objectClass	labeled URI sap Name service Hint
eService	objectClass startMode	startMode startup Parameters sapPtr serviceName

Table 16-5 LDAP attributes

Attribute	Table name	Type	Size	Access	Value
labeledURI	labeledURI	case Exact String	100	normal	multiple
sap Name	sapName	case Ignore String	256	normal	single
sap Ptr	sapPtr	DN	1000	normal	multiple
service Hint	serviceHint	DN	1000	normal	single
service Name	serviceName	case Ignore String	256	normal	single
start Mode	startMode	case Exact String	10	normal	single
startup Parameters	startup Parameters	case Exact String	256	normal	single

16.6.3 Administration considerations

Security server operator commands are shown here with the new parameters for the `MODIFY SKRBKDC` command as follows:

```
F SKRBKDC,parameters
```

When entering this command, the parameters specified are the commands to be executed by the security server.

Parameter commands

The commands you can enter as parameters are as follows:

DISPLAY LEVEL Displays the current service level of the Kerberos security server. This option may be abbreviated `D LEVEL`.

DISPLAY CRYPTO Displays the available encryption types, whether hardware cryptographic support is available, and whether the encryption type can be used for application data. This option may be abbreviated `D CRYPTO`.

DISPLAY XCF Displays the status of all instances of the SKRBKDC started task in the sysplex. This command can be abbreviated as `D XCF`.

DISPLAY CREDS,*owner,date* Displays all credentials data space allocations for a user that were created before the specified date. All data space allocations for a user are displayed if the date is omitted. All data space allocations are displayed if no owner is specified. A date can be specified without specifying an owner by using two successive commas. A maximum of 252 allocations can be displayed. The date is specified as *yyyy.ddd*.

This command can be abbreviated as **D CREDS,owner,date**.

16.7 New security server commands

This section presents new Network Authentication Service for z/OS commands. It provides the format, options, usage, and examples for each command. It also shows the changes in the usage comments of the **keytab** command.

The commands are installed in /usr/lpp/skrb/bin. In order to use these commands, you must update the PATH environment variable to place /usr/lpp/skrb/bin before /bin in the search order or else you must use the fully-qualified command name.

16.7.1 The kadmin command

The **kadmin** command is used to manage entries in the Kerberos database. It prompts you to enter one or more subcommands. The **kadmin** command can be used with any Kerberos administration server supporting Version 2 of the Kerberos administration protocol. The command has the following format:

```
kadmin [-r realm ][-p principal ][-k keytab ][-w password ][-A ][-e ]
```

Where:

-r realm Specifies the Kerberos administration realm. If this option is not specified, the realm is obtained from the principal name. This option is meaningful only if the administration server supports multiple realms.

-p principal Specifies the administrator principal. If this option is not specified, the string **/admin** is appended to the principal name obtained from the default credentials cache. If there is no credentials cache, the string **/admin** is appended to the name obtained from the USER environment variable, or, if the USER environment variable is not defined, it is appended to the name obtained from the **getpwuid()** function. The local realm is used if an explicit realm is not part of the principal name.

The principal name is **host/***host-name* unless the **-p** option is specified. The *host-name* is the primary host name for the local system.

-k keytab Specifies the key table containing the password for the administrator principal. The user is prompted to enter the password if neither the **-k** nor the **-w** option is specified.

-w password Specifies the password for the administrator principal. The user is prompted to enter the password if neither the **-k** nor the **-w** option is specified.

-A Specifies that the initial ticket used by the **kadmin** command does not contain a list of client addresses. If this option is not specified, the ticket contains the local host address list. When an initial ticket contains an address list, it can be used only from one of the addresses in the address list.

-e Echoes each command line to stdout. This is useful when stdout is redirected to a file.

Note: Subcommand options start with a minus (-) character and principal attributes start with a plus (+) character or a minus (-) character.

The **kadmin** command imposes no other restrictions on the characters used in names or passwords, although it is recommended that you do not use any of the EBCDIC variant characters. The Kerberos administration server may impose additional restrictions.

Time units

You can use time units such as dates, which are displayed as *day-of-week, month, day-of-month, hour:minute:second, timezone,* or *year* using the local time zone, as specified by the TZ environment variable. Durations are displayed as *days-hours:minutes:seconds*.

The **kadmin** command supports a number of date and duration formats and some examples are as follows:

```
"15 minutes" - "7 days" - "1 month" - "2 hours" - "400000 seconds" - "next year" - "this
Monday"
```

See *z/OS SecureWay Security Server Network Authentication Service Administration*, SC24-5926 for more details and examples.

Subcommands

The following subcommand descriptions assume the administration server is using the standard MIT Kerberos database for the registry. Other database implementations may not support all of the subcommand options and attributes.

Principal-related commands

> **Note:** For the subcommands that follow:
>
> - Name specifies a Kerberos principal
>
> There is a long list of options that can be used in defining a principal, such as the types of tickets it can use, what services it can provide, what encryption types are supported for this principal, and what pre-authentication steps might be required

See *z/OS SecureWay Security Server Network Authentication Service Administration,* SC24-5926 for the attributes can be displayed by these subcommands.

The following subcommands are supported:

help [*subcommand***]**
The **help** subcommand displays the command syntax for the specified subcommand. If no subcommand name is specified, the available subcommands are displayed.

list_principals [*expression***]**
The **list_principals** (also known as **listprincs**) subcommand lists all of the principals in the Kerberos database that match the specified search expression. If no search expression is provided, all principals are listed. You must have LIST authority.

get_principal *name*
The **get_principal** (also known as **getprinc**) subcommand displays information for a single principal entry. You must have GET authority, or the principal entry must be your own entry.

add_principal [*options***][***attributes***]** *name*
The **add_principal** (also known as **addprinc**) subcommand adds a new principal entry to the Kerberos database. The options and attributes may be specified before or after the principal name and may be entered in any order. You must have ADD authority.

`delete_principal` *`name`*
The **`delete_principal`** (also known as **`delprinc`**) subcommand deletes a principal entry from the Kerberos database. You must have DELETE authority.

`modify_principal [`*`options`***`][`***`attributes`***`]`** *`name`*
The **`modify_principal`** (also known as **`modprinc`**) subcommand modifies an existing principal entry in the Kerberos database. The options and attributes may be specified before or after the principal name and may be entered in any order. You must have MODIFY authority.

`change_password [-randkey | -pw` *`password`***`]`** *`name`*
The **`change_password`** (also known as **`cpw`**) subcommand changes the password for a principal. You must have CHANGEPW authority, or the principal entry must be your own entry.

`rename_principal` *`oldname newname`*
The **`rename_principal`** (also known as **`renprinc`**) subcommand changes the name of a principal entry in the Kerberos database. You must have both ADD and DELETE authority.

Policy-related commands

Note: Policy is associated with a password. It specifies characteristics such as the password lifetime, length, number of character classes that must be present, and number of passwords kept in the password history. Passwords in the password history cannot be reused.

See *z/OS SecureWay Security Server Network Authentication Service Administration*, SC24-5926 for the following options that are supported for the policy related subcommand.‘

`list_policies [`*`expression`***`]`**
The **`list_policies`** (also known as **`listpols`**) subcommand lists all of the policies in the Kerberos database that match the specified search expression. All policies are listed if no search expression is provided. You must have LIST authority.

`get_policy` *`name`*
The **`get_policy`** (also known as **`getpol`**) subcommand displays information for a single policy entry. You must have GET authority or the policy must be associated with your own principal entry.

`add_policy [`*`options`***`]`** *`name`*
The **`add_policy`** (also known as **`addpol`**) subcommand adds a new policy to the Kerberos database. The options may be specified before or after the policy name and may be specified in any order. You must have ADD authority.

`modify_policy [`*`options`***`]`** *`name`*
The **`modify_policy`** (also known as **`modpol`**) subcommand modifies an existing policy in the Kerberos database. The options may be specified before or after the policy name and may be specified in any order. You must have MODIFY authority.

`delete_policy` *`name`*
The **`delete_policy`** (also known as **`delpol`**) subcommand deletes a policy entry from the Kerberos database. You must have DELETE authority.

add_key [[-keytab|-k] *keytab_name***]** *principal_name*
The **add_key** (also known as **ktadd**) subcommand generates a set of random encryption keys for the named principal and then adds the generated keys to the specified key table. The default key table is used if the **-keytab** option is not specified. A key table name prefix of ‘FILE‘ is changed to ‘FILE‘ because the **add_key** subcommand must update the key table.

16.7.2 The keytab command

The **keytab** command manges a key table and is used to add or delete a key from a key table or to display the entries in a key table. The **keytab** command creates key table entries for each unique key type when adding a key version to the key table. Key table entries are not created for all supported encryption types because some of the encryption types share the same key type. When retrieving a key from the key table, the Kerberos runtime selects the appropriate key based upon the requested encryption type.

Table 16-6 displays the key types and associated encryption types that are supported.

Table 16-6 Key types and associated encryption types

Key types	Encryption types
56-bit DES	des-cbc-crc (ENCTYPE_DES_CBC_CRC) des-cbc-md4 (ENCTYPE_DES_CBC_MD4) des-cbc-md5 (ENCTYPE_DES_CBC_MD5) des-cbc-raw (ENCTYPE_DES_CBC_RAW)
56-bit DES with key derivation	des-hmac-sha1 (ENCTYPE_DES_HMAC_SHA1)
168-bit DES with key derivation	des3-cbc-sha1 (ENCTYPE_DES3_CBC_SHA1) des3-cbc-raw (ENCTYPE_DES3_CBC_RAW)

16.7.3 The kpasswd command

The **kpasswd** command changes the password for a Kerberos principal using the password change service. You must supply the current password for the principal as well as the new password. The password change server applies any applicable password policy rules to the new password before changing the password. The command is issued as follows:

```
kpasswd [principal]
```

The *principal* option specifies the principal whose password is to be changed. The principal is obtained from the default credentials cache if the principal is not specified on the command line.

Note: You may not change the password for a ticket-granting service principal (**krbtgt/***realm*) using the **kpasswd** command.

16.7.4 The kvno command

The **kvno** command displays the current key version number for a principal and is issued as follows:

```
kvno [principal]
```

The *principal* option specifies the principal whose current key version number is to be displayed. The principal is obtained from the default credentials cache if the principal is not specified on the command line.

16.8 New status codes and messages

See *z/OS SecureWay Security Server Network Authentication Service Administration*, SC24-5926 for information about new status codes and messages.

16.9 SKRBKDC automatic start or restart

Since the normal operating mode is to run the SKRBKDC start task on each system in the sysplex, the SKRBKDC started task registers with Automatic Restart Management (ARM) to be restarted only if the started task fails and not if the current system fails. The TERMTYPE parameter of the ARM policy can be used to override this registration if the SKRBKDC started task is not running on another system in the sysplex and thus should be restarted on a backup system in the sysplex when the current system fails.

Note: Dynamic virtual IP addressing (DVIPA) should be used in this case and the DVIPA backup system must be the same system which will be used to restart skrbkdc.

16.9.1 ARM automatic restart

The security server has now registers with Automatic Restart Management (ARM) on a specified system. The security server is automatically restarted if it fails unexpectedly. The security server is not restarted if it detects errors and stops. The ARM element type is SYSKERB and the ARM element name is EUVFKDC_system-name. The ARM policy can be used to override the default registration values if needed.

16.10 KMIGRATE UNIX utility

The kmigrate utility is a tool for migrating existing DCE and MVS users to a Kerberos registry managed by an OS/390 SecureWay Network Authentication and Privacy Service server. The intent is to ease the migration of users who are currently using Kerberos services, with or without DCE, to the IBM Kerberos server, and to ease the initial population of a Kerberos registry from an existing RACF registry. This tool analyzes the existing RACF and DCE registries and provides a list of suggested RACF commands to prime the Kerberos registry with Kerberos principals. Where necessary, this tool will also suggest MVS userids for these principals. This tool will provide suggested Kerberos principals to correspond to DCE principals and MVS userids. The output of the **kmigrate** command is a file that can be run as a CLIST to make the RACF changes. No changes will be made to the RACF registry until the customer has a chance to review these suggestions and make changes.

16.10.1 The kmigrate command

The **kmigrate** command is run from the UNIX System Services shell. This command generates a suggested starter set for a Kerberos registry using the existing MVS userids on the target system, and also assists in the migration of users from a DCE registry to a Kerberos registry. The DCE principal may or may not be cross-linked to MVS userids and has the following format:

```
kmigrate -r racf_db_file [-d dce_principal_file] [-k kerberos_prefix] [-n mvs] [-o
output_file] [-e error_file]
```

The parameters and options are specified as follows:

-r racf_db_file Specifies the name of a file or data set containing the output of RACF database unload (IRRDBU00). This file must contain records for users, DCE segments, and Kerberos segments, for the system whose registry is to be primed. Other records will be ignored. This file is required.

-d dce_principal_file Specifies the name of a file or data set containing a list of DCE principals. One way to generate such a list is to issue the command `dcecp -c principal catalog -simplename` and pipe the results to a file. This file is optional. You may omit this file if you have no DCE cell, if you wish to handle defining new userids for DCE users at another time, or if all of the DCE principals you wish to define in the Kerberos registry are cross-linked to MVS userids.

-k kerberos_prefix Specifies a prefix of 1 to 3 characters to be used in generating MVS userids to go with Kerberos principals. This prefix will be combined with a 4- to 6-digit number to create a 7-character userid. The default prefix is KRB.

-n mvs Specifies that MVS userids without DCE segments are not to be included in the output file. This causes `ADDUSER` and `ALTUSER` commands to be generated for DCE users only. Commands will be generated for DCE principals, whether or not they have been cross-linked to MVS userids.

-o output_file Specifies the name of a file or data set to contain the output. The default is stdout.

-e error_file Specifies the name of a file or data set to contain error data. This file will contain information about MVS userids that cannot easily be assigned to Kerberos principals because the Kerberos principal already exists, and about DCE principals that cannot be easily resolved uniquely to either MVS userids or Kerberos principals because either the MVS userid already exists or a Kerberos principal already exists. The default is `stderr`.

Command output

This command generates an input file, to be used by a RACF administrator to define Kerberos principals for OS/390 Kerberos, and is placed in the file indicated by the *-o* option.

Note: MVS userids and Kerberos principals need to be unique. In the event that a duplicate Kerberos principal or a duplicate MVS userid would be generated, a message will be output to the file indicated by the *-e* option.

The commands that are generated in the input file are as follows:

- For each MVS user, an `ALTUSER` command will be output to generate a Kerberos segment. If the user has a DCE segment, the Kerberos principal will be the same as the DCE principal. If the user does not have a DCE segment, the Kerberos principal will be the same as the MVS userid.
- No output will be generated for MVS userids that already have Kerberos segments.
- For each DCE user that is not an MVS user, an `ADDUSER` command will be output to generate an MVS user with a Kerberos segment. No other options will be used for `ADDUSER`. If these are used "as is," the system default group and password will be assigned. The MVS userid will be set to the prefix specified by the -k option (default KRB) followed by a 4-digit to 6-digit number to create a 7-character userid. The Kerberos principal will be set to the DCE principal. The following architected DCE principals will not

be included in the output: hosts/*, nobody, root, daemon, sys, bin, uucp, who, mail, tcb, dce-ptgt, and dce-rgy.

- For all existing DCE users: if the DCE principal name is too long to be used as a Kerberos principal (over 240 characters), the last 240 characters will be used to generate the Kerberos principal.

After the **kmigrate** command is run, the user should examine the output file and the error file. These can be edited as desired to add additional information to **ADDUSER** and **ALTUSER** commands and also to modify userids and principals. Conflicts indicated in the error file can be resolved by hand and additional **ADDUSER** and **ALTUSER** commands can be added. Records can also be deleted. When all editing is complete, the list can be run in TSO using the EXEC command.

Command examples

To generate the **ADDUSER** and **ALTUSER** list, where the IRRDBU00 output is in irrdbu00.txt and the DCE principal list is in dcecp.txt, with output to user.txt:

```
kmigrate -r irrdbu00.txt -d dcecp.txt -o user.txt
```

To generate the **ADDUSER** and **ALTUSER** list, where the IRRDBU00 output is in data set G085573.PRIVATE.SUSVT5.RACF and the DCE principal list is in dcecp.txt, with output to data set G085573.PRIVATE.SUSVT5.KERB:

```
kmigrate -r "//'G085573.PRIVATE.SUSVT5.RACF'" -d dcecp.txt -o \
"//'G085573.PRIVATE.SUSVT5.KERB'"
```

To generate the same list, but using “KR” as a userid prefix and logging errors in error.txt:

```
kmigrate -r irrdbu00.txt -d dcecp.txt -o user.txt -e error.txt -k KR
```

> **Note:** Userids generated by the above command will have the format KRnnnnn (7 characters: 2-character prefix, 5 digits).

Sample output

Figure 16-3 on page 294 is an abbreviated sample of output from the **kmigrate** command. Input was taken from the RACF registry and the DCE registry. All other options were allowed to default. Note that users G9VWPF and SUDFS3 already have DCE principals.

```
ADDUSER KRB0001 KERB(KERBNAME('kadmin/changepw'))
ADDUSER KRB0002 KERB(KERBNAME('kadmin/admin'))
ADDUSER KRB0003 KERB(KERBNAME('krbtgt/dcesvt2.krbsvt52.ibm.com'))
ALTUSER XXX KERB(KERBNAME('XXX'))
ALTUSER WEBSRV KERB(KERBNAME('WEBSRV'))
ALTUSER WEBADM KERB(KERBNAME('WEBADM'))
ALTUSER USER02 KERB(KERBNAME('USER02'))
ALTUSER USER01 KERB(KERBNAME('USER01'))
ALTUSER SUDFS3 KERB(KERBNAME('cell_admin'))
ALTUSER SUDFS2 KERB(KERBNAME('SUDFS2'))
ALTUSER SUDFS1 KERB(KERBNAME('SUDFS1'))
ALTUSER SKRBKDC KERB(KERBNAME('SKRBKDC'))
ALTUSER SERVICE KERB(KERBNAME('SERVICE'))
ALTUSER SERVER KERB(KERBNAME('SERVER'))
ALTUSER SECADMIN KERB(KERBNAME('SECADMIN'))
ALTUSER OMVSKERN KERB(KERBNAME('OMVSKERN'))
ALTUSER ODEROOT KERB(KERBNAME('ODEROOT'))
ALTUSER NFS KERB(KERBNAME('NFS'))
ALTUSER MVSSTC KERB(KERBNAME('MVSSTC'))
ALTUSER IBMUSER KERB(KERBNAME('IBMUSER'))
ALTUSER G9VWPF KERB(KERBNAME('g9vwpf'))
ALTUSER CDSD KERB(KERBNAME('CDSD'))
ALTUSER CDSCLRK KERB(KERBNAME('CDSCLRK'))
ALTUSER CDSADV KERB(KERBNAME('CDSADV'))
ALTUSER CANCEL KERB(KERBNAME('CANCEL'))
ALTUSER BPXROOT KERB(KERBNAME('BPXROOT'))
```

Figure 16-3 Sample output from the **`kmigrate`** *command*

16.10.2 Procedure for migrating a DCE registry to Kerberos

This following steps will prime an OS/390 Kerberos registry with principals taken from an existing DCE cell and also existing MVS userids.

1. Prepare the IBM Kerberos environment. To add Kerberos segments with principals, RACF must be able to determine the name of the local Kerberos realm. The IBM Kerberos server should be installed and configured according to the instructions in the Program Directory and in *z/OS SecureWay Security Server Network Authentication Service Administration*, SC24-5926.
2. Run the RACF Database Unload job (IRRDBU00) according to the instructions in the *RACF System Administrator's Guide*, and save the output. This should be run on the system where the Kerberos server will reside. If desired, run the resulting file through a sort program to extract records of type 0200 (User basic), 0290 (DCE user), and 02D0 (Kerberos user). This additional step is not required but it will reduce the size of the file that is input to the kmigrate program. Optionally, copy the resulting file to an HFS directory using the **`oput`** command.
3. Optional step for DCE customers: Run the **`dcecp`** command **`dcecp -c principal catalog -simplename`** and pipe the output to a file. Copy the file to an HFS directory.
4. Optional step for DCE customers: Determine the userid prefix to be used for DCE principals who are not already cross-linked to MVS userids. This prefix will be used to generate a userid for each principal. It should be 1 to 3 characters. By default, userids will be of the format KRBnnnn where nnnn is a 4-digit number.
5. Run the **`kmigrate`** command.

6. Examine the resulting output file and make changes as desired. If the output is in an HFS file, use the TSO **oget** command to move it to a data set.
7. Run the output file as a CLIST, trapping output so you can check for errors. This will issue the RACF commands to create and alter MVS userids to add Kerberos segments with the specified principals.

   Note: RACF "special" authority is required in order to execute the **ADDUSER** and **ALTUSER** commands.
8. Examine the RACF log files for errors.

Each modified MVS userid will need to have its password changed to generate a Kerberos key for the principal and enable participation in Kerberos authentication.

As automatically generated, each new MVS userid has a bare-bones definition. The password will start out expired, and no Kerberos key will be generated for the user. To generate a valid password and Kerberos key, the user can logon through **rlogin** and change the password, or the administrator can change the password to a non-expired value. A Kerberos key will be generated for the principal when the password is changed.

A

Sample GRS exit ISGNQXIT

This appendix contains sample exit code that replaces sample exit ISGGREX0 that was available from the ITSO beginning in OS/390 Release 6.

This appendix contains the following:

- Sample exit code for exit ISGNQXIT

Sample ISGNQXIT exit

This sample exit code is provided to support shared DASD with z/OS and OS/390 systems across GRS-plexes.

Attention: To obtain the exit, you must download **ISGNQXIT.ZIP**. The .ZIP file contains:

- ISGNQXIT.ASM
- ISGNQXIT.README

Please read the following carefully, (This is the README file).

1) The file **ISGNQXIT.ASM** must be uploaded in BINARY with LRECL=80.

2) Response to requests to support type PATTERN

The ISGNQXIT exit has been modified to support type PATTERN for the RNL Exclusion table scan. A customer using the exit can now use type PATTERN definitions in all RNL tables, including Conversion. The documentation in the IBM Redbook SG24-6235 says PATTERN is not supported; with this new version, it is supported for the RNL Exclusion table scan.

Example:

A customer can use the following RNL Conversion definition to convert all Reserves:

```
RNLDEF RNL(CON) TYPE(PATTERN)
 QNAME(*)                         /*  TEST FOR GRS EXIT        */
```

The entries in RNL Conversion Table for special QNAME HWRESERV with RNAME being a volser (used by the exit to guarantee HW Reserve for cross-sysplex volume share), can ONLY be type SPECIFIC or GENERIC, because using type PATTERN for these definitions may generate confusion The exit, during the RNL Conversion table scan, skips the type PATTERN entries, searches for qname HWRESERV and then compares the volser using the length of the rname. Again, type PATTERN is not supported if you have QNAME(HWRESERV).

Examples:

```
RNLDEF RNL(CON) TYPE(SPECIFIC)
QNAME(HWRESERV)               /*volser SBOX23 is shared cross sysplex*/
RNAME(SBOX23)

RNLDEF RNL(CON) TYPE(GENERIC)
QNAME(HWRESERV)               /*volsers starting with BOOK are shared*/
RNAME(BOOK)
```

The source code, changed on May 23, 2002, can be downloaded via the Internet by doing the following:

Attention: Sample source code/procedures are available in softcopy on the Internet from the Redbooks Web server.

Point your browser to:

`ftp://www.redbooks.ibm.com/redbooks/SG246235/`

Note: SG246235 must be upper case (the SG)

Alternatively, you can go to:

`http://www.redbooks.ibm.com`

and select REDBOOKS ONLINE, and then Additional Materials.

```
//MERONI   JOB (999,POK),CLASS=A,                                       00010000
//         MSGLEVEL=(1,1),MSGCLASS=X,                                   00020000
//         NOTIFY=&SYSUID                                               00030000
//ASMH   EXEC PGM=ASMA90,REGION=1024K,PARM='DECK,XREF(SHORT)'           00040000
//SYSPRINT  DD SYSOUT=*                                                 00050000
//SYSUT1    DD UNIT=SYSDA,SPACE=(CYL,(5,5)),DISP=(NEW,DELETE)           00060000
//SYSPUNCH  DD DSN=&OBJ(AUDIT),DISP=(,PASS,DELETE),UNIT=SYSDA,          00070000
//             SPACE=(TRK,(1,5,5))                                      00080000
//SYSPUNCH  DD DUMMY                                                    00090000
//SYSGO     DD DUMMY                                                    00100000
//SYSLIN    DD DUMMY                                                    00110000
//SYSLIB    DD DSN=SYS1.MACLIB,DISP=SHR                                 00120000
//          DD DSN=SYS1.MODGEN,DISP=SHR                                 00130000
//SYSIN DD *                                                            00140000
         TITLE 'ISGNQXIT - ENQ/DEQ EXIT ROUTINE'                        00150000
ISGNQXIT CSECT                                                          00160000
*/*  START OF SPECIFICATIONS ****                                       00170000
*-----------------------------------------------------------------*     00180000
* Cross Sysplex DASD sharing                                      *     00190000
* Always H/W RESERVE Volumes shared cross GRS complexes           *     00200000
*-----------------------------------------------------------------*     00210000
*                                                                 *     00220000
* It is mandatory to support SHARED DASD with OS/390 Systems      *     00230000
* cross GRS-plexes that all RESERVE requests that address these   *     00240000
* VOLUMES result in an H/W RESERVE whatever the RESOURCE NAME is. *     00250000
*                                                                 *     00260000
* The same RESOURCES, that address other VOLUMES, should have the *     00270000
* possibility to be filtered through the CONVERSION RNLs and have *     00280000
* the H/W RESERVE eliminated.                                     *     00290000
*                                                                 *     00300000
* Using ISGNQXIT GRS exit and by adding to the CONVERSION table   *     00310000
* a QNAME not used by the System and RNAMEs that identify the     *     00320000
* Volumes to share outside GRS, the RESERVE requests for the      *     00330000
* Volumes included in the following definition will always result *     00340000
* in a H/W RESERVE.                                               *     00350000
*                                                                 *     00360000
*      RNLDEF RNL(CON) TYPE(SPECIFIC/generic)                     *     00370000
*      QNAME(HWRESERV)                                            *     00380000
*      RNAME(VOLSER/volser-prefix)                                *     00390000
*                                                                 *     00400000
* LOGIC:                                                          *     00410000
*                                                                 *     00420000
* ISGNQXIT  receives control for all RESERVE/DEQ  requests        *     00430000
* (SYSTEMS+H/W RESERVE), locates the 'VOLSER'  and checks if the  *     00440000
* resource 'HWRESERV' 'VOLSER' is present in the conversion       *     00450000
* table. If found the RNL processing is bypassed, if not normal   *     00460000
* RNL scan is perormed.                                           *     00470000
*                                                                 *     00480000
* Because the RNL CONVERSION table is used, any change to the     *     00490000
* table can be activated through OS/390 operator command:         *     00500000
* SET GRSRNL=xx.                                                  *     00510000
*                                                                 *     00520000
* The name 'HWRESERV' is hard-coded and is defines at label       *     00530000
* HRDWNAME in the ISGNQXIT example.                               *     00540000
*                                                                 *     00550000
* NOTE: the VOLUME(S) behind the QNAME(HWRESERV) should not be    *     00560000
* dynamically changed unless the device(s) is(are) offline.       *     00570000
*                                                                 *     00580000
* WILDCARDING IS NOT SUPPORTED FOR HWRESERV RNL ENTRIES.          *     00590000
*                                                                 *     00600000
*-----------------------------------------------------------------*     00610000
*                 CONVERSION LIST EXAMPLE                         *     00620000
*-----------------------------------------------------------------*     00630000
*                                                                 *     00640000
* RNLDEF RNL(CON) TYPE(GENERIC)                                   *     00650000
* QNAME(SYSVTOC)                                                  *     00660000
*                                                                 *     00670000
* RNLDEF RNL(CON) TYPE(GENERIC)                                   *     00680000
* QNAME(SYSIGGV2)                                                 *     00690000
*                                                                 *     00700000
* RNLDEF RNL(CON) TYPE(SPECIFIC)                                  *     00710000
* QNAME(HWRESERV)                     /*SPECIAL NAME*/            *     00720000
* RNAME(XA9RES)                       /*ALWAYS RESERV  XA9RES*/   *     00730000
*                                                                 *     00740000
* RNLDEF RNL(CON) TYPE(GENERIC)                                   *     00750000
* QNAME(HWRESERV)                     /*SPECIAL NAME*/            *     00760000
* RNAME(CIX)                          /*ALWAYS RESERV VOLUMES*/   *     00770000
*                                     /*BEGINNING WITH CIX   */   *     00780000
*                                                                 *     00790000
***                         END OF                                *     00800000
***   RESERVE CONVERSION RESOURCE NAME LIST - SAMPLE              *     00810000
*******************************************************************     00820000
* RESTRICTION:                                                    *     00830000
*                                                                 *     00840000
* 1-The exit does not propagate cross Sysplex ENQ requests with   *     00850000
* scope=SYSTEMS, only guarantees that the H/W Reserve is issued   *     00860000
* for Volumes behind qname HWRESERV in RNL Conversion table.      *     00870000
*                                                                 *     00880000
* 2-Type PATTERN is not supported for HWRESERV RNL entries        *     00890000
*                                                                 *     00900000
*Note-Restriction of type PATTERN not supported for EXCLUSION RNL *     00901002
*     HAS BEEN REMOVED                                            *     00901102
*                                                                 *     00902000
*******************************************************************     00903000
* COMPATIBILITY:                                                  *     00904000
*                                                                 *     00905000
* ISGNQXIT is compatible with ISGGREX1 ITSO exit provided for     *     00906000
* OS/390 Systems without GRS WILDCARD support.                    *     00907000
*                                                                 *     00908000
*******************************************************************     00909000
*                                                                 *     00910000
                                                                        00920000
*******************************************************************     00930000
**EXIT INSTALLATION                                               *     00940000
```

```
*                                                                  *
* THE EXIT CAN BE INSTALLED WITH ONE OF THE FOLLOWING METHODS      *
*                                                                  *
* 1.WITH A PROGRAM USING CSVDYNEX MACRO                            *
*                                                                  *
* 2.WITH SETPROG EXIT OPERATOR COMMAND                             *
*   EX. SETPROG EXIT,ADD,EX=ISGNQXIT,MOD=ISGNQXIT,                 *
*       DSN=SYS1.USER.LINKLIB                                      *
*                                                                  *
* 3.USING EXIT STATEMENT OF THE PROGXX PARMLIB MEMBER              *
*       EXIT ADD                                                   *
*            EXITNAME(ISGNQXIT)                                    *
*            MODNAME(ISGNQXIT)                                     *
*            STATE=ACTIVE                                          *
*            DSNAME(SYS1.USER.LINKLIB)                             *
*                                                                  *
*                                                                  *
*NOTE: Recommnded procedure to activate the exit after IPL:       *
*      1.Activate the exit in all Systems sharing ALWAYS RESERVE  *
*        VOLUMES                                                   *
*      2.Activate the RNLS in all Systems sharing ALWAYS RESERVE  *
*        VOLUMES. The Volume(s) behind the QNAME(HWRESERV) should *
*        not be dynamically added/removed unless the devices are  *
*        OFFLINE                                                   *
********************************************************************
       EJECT
********************************************************************
*
*         REGISTERS-SAVED =
*                           R0 - R12, R14, R15
*
*         REGISTER-USAGE =
*                 R0     = CONTAINS RNL ADDRESS ON ENTRY. CONTAINS
*                            RETURN CODE WHILE SCANNING LISTS
*                 R1     = ADDRESS OF A ENQPL ON ENTRY
*                 R2     = USED TO ADDRESS RNLES
*                 R3-R5  = USAGE UNPREDICTABLE
*                 R6     = UCB POINTER
*                 R7-R12 = USAGE UNPREDICTABLE
*                 R13    = SAVE AREA ADDRESS
*                 R14    = RETURN ADDRESS
*                 R15    = ENTRY POINT ADDRESS/BASE REGISTER.
*                            UPON EXIT, RETURN CODE.
*
*         REGISTERS-RESTORED =
*                              R0 - R12, R14
*
*         RETURN-CODES =
*                        R15 = 0 -
*                              4 - NOT USED
*
*       EXIT-ERROR = NONE
*
*       WAIT-STATE-CODES = NONE
*
*     EXTERNAL-REFERENCES = NONE.
*
*       ROUTINES = NONE.
*
*       DATA-AREAS = NONE
```

```
*
*       CONTROL-BLOCKS =
*                        CVT       R
*                        ENQPL     R/W
*                        GVT       R
*
*     TABLES =
*              1. RESERVE CONVERSION RNL        (ISGGCRNL)
*
*       SERIALIZATION = NONE.
*
*01*  CHANGE-ACTIVITY =  INITIAL RELEASE 1.0                   08/01
*
*02*  Bypass exit code for deq requests to improve    *DEQ     03/02
*     performance                                     *DEQ
*
*03*  Move BASEREG initialization after registers saving       04/02
*
*04*  RELEASE 2.0 -                                            05/02
*     Type PATTERN support for RNL EXCL entries
*
*     MESSAGES = NONE.
*
*     ABEND-CODES = NONE.
*
**** END OF SPECIFICATIONS **                                        */
         EJECT
***********************************************************************
*                                                                     *
*        REGISTER ASSIGNMENTS                                         *
*                                                                     *
***********************************************************************
         SPACE
ZERO     EQU   0
ENQPL    EQU   1                       ADDRESS OF THE ENQ PLIST
RNLEPTR  EQU   2                       POINTER TO AN RNL ENTRY (RNLE)
FLENRNLE EQU   4                       LENGTH OF FIXED PART OF AN RNLE
RNAMELEN EQU   5                       LENGTH OF RNAME
BASEREG  EQU   11                      BASE REGISTER
WORKREG  EQU   12                      WORK REGISTER
REG13    EQU   13                      SAVE AREA ADDRESS
REG14    EQU   14                      RETURN ADDRESS
REG15    EQU   15                      RET CODE
*
R0       EQU   0
R1       EQU   1                       ADDRESS OF ENQPL
R2       EQU   2                       POINTER TO AN RNLE
R3       EQU   3                       CVT, GVT
R4       EQU   4
R5       EQU   5                       RNLE LEN, PATTERN SCAN
R6       EQU   6                       RETURN REG, PATTERN SCAN
R7       EQU   7                       Q-RNAME LEN, PATTERN SCAN
R8       EQU   8                       RNLE INCREMENT POINTER
R9       EQU   9                       Q_RNAME INCREMENT POINTER
R10      EQU   10                      Q_RNAME POINTER, PATTERN SCAN
R11      EQU   11                      BASE REG
R12      EQU   12                      WORK REG
R13      EQU   13                      SAVE AREA POINTER
R14      EQU   14                      RETURN ADDRESS
R15      EQU   15
```

```
       EJECT                                                            02150000
*********************************************************************** 02160000
*                                                                     * 02170000
*        LOGIC FLOW FOR ISGNQXIT                                      * 02180000
*                                                                     * 02190000
*********************************************************************** 02200000
*/*                                                                  */ 02210000
*/*++ 'ISGNQXIT': ENTRY TO ENQ DEQ EXIT                              */ 02220000
*/*++ ESTABLISH ADDRESSABILITY                                       */ 02230000
*/*++ IF DEQ                            --> RETURN                   */ 02240000
*/*++ IF THE ENQPL UCB POINTER NOT ZERO --> RETURN                   */ 02250000
*/*                                                                  */ 02260000
*/*++   LOCATE RNL EXCLUSION  TABLE POINTER                          */ 02270000
*/*++     DO WHILE MATCH NOT FOUND AND NOT LAST ENTRY IN THE RNL     */ 02280000
*/*++       IF RNL MAJOR + MINOR MATCH --> RETURN                    */ 02290000
*/*++     ENDDO  (REPEAT SEQUENCE UNTIL MATCH FOUND OR END OF RNL)   */ 02300000
*/*                                                                  */ 02310000
*/*++   LOCATE RNL CONVERSION TABLE POINTER                          */ 02320000
*/*++     DO WHILE MATCH NOT FOUND AND NOT LAST ENTRY IN THE RNL     */ 02330000
*/*++       IF RNL MAJOR NAME IS HWRESERV                            */ 02340000
*/*++         IF RNLE MINOR NAME EQUALS ENXP VOLSER                  */ 02350000
*/*++           SET BYPASS RNL PROCESSING FLAG                       */ 02360000
*/*++         ENDIF    (END OF RNAME COMPARISON)                     */ 02370000
*/*++       ENDIF  (END OF CHECK FOR RNL MAJOR NAME HWRESERV         */ 02380000
*/*++     ENDDO  (REPEAT SEQUENCE UNTIL MATCH FOUND OR END OF RNL)   */ 02390000
*/*++    ENDIF  (END OF ENXP  PROCESSING)                            */ 02400000
*/*++ RETURN                                                         */ 02410000
*/*++ END 'ISGNQXIT'                                                 */ 02420000
*/*                                                                  */ 02430000
*/******************************************************************/ 02440000
         SPACE 3                                                        02450000
ISGNQXIT AMODE 31                                                       02460000
ISGNQXIT RMODE ANY                                                      02470000
         MODID BR=NO,MODLBL=ITSOQXIT                                    02480000
ISGCVXIT DS    OH                                                       02490000
*                                                                       02500000
*3-----  LR    BASEREG,R15                                              02510000
         ENTRY ISGCVXIT                                                 02520000
         SPACE                                                          02530000
         STM   14,12,12(13)             SAVE ENTRY REGS                 02540000
         LR    BASEREG,R15                                              02550000
         USING ISGCVXIT,BASEREG                                         02560000
         USING NQXP,ENQPL               ENQ PLIST ADDRESSABILITY        02570000
*                                                                       02580000
*2----- PMS  48220------                                                02590000
         TM    NQXP_STATEFLAGS1,NQXP_SF1_ENQ  RETURN IF DEQ   *DEQ      02600000
         BZ    MODEXIT                                        *DEQ      02610000
*2----- PMS  48220------                                                02620000
*                                                                       02630000
         ICM   R6,15,NQXP_OP_UCB@       UCB POITER                      02640003
         BZ    MODEXIT                  No Reserve, exit                02650003
*                                       LOCATE RNL CONV                 02660000
         L     R3,FLCCVT-PSA            CVT                             02670000
         USING CVT,R3                                                   02680000
         L     R3,CVTGVT                GET GRS VECTOR                  02690000
         DROP  3                                                        02700000
         USING GVT,R3                                                   02710000
         TM    GVTVFLAG,GVTVCRNL        RESERVE CONVERSION INVALID      02720000
         BO    MODEXIT                  YES, EXIT                       02730000
*                                                                       02740000
```

```
         TM    GVTVFLAG,GVTVERNL        SYSTEMS EXCLUSION  INVALID      02750000
         BO    MODEXIT                  YES, EXIT                       02760000
         ICM   RNLEPTR,15,GVTSERNL      SYSTEMS EXCLUSION  RNL          02770000
         BZ    MODEXIT                  NO RNL EXCLUSION                02780000
*                                                                       02790000
         L     R14,COMPE                RETURN ADDRESS                  02800000
*                                                                       02810000
         USING RNLE,RNLEPTR                                             02820000
*------------------------------------------------------------------*    02830000
* EXCLUSION RNL LIST SEARCH                                             02840000
*------------------------------------------------------------------*    02850000
         SPACE                                                          02860000
COMPRNLE EQU   *                                                        02870000
         TM    RNLEFLGS,RNLELAST       AT END OF THE RNL ?              02880000
         BO    NOMATCH                 YES => EXCL SCAN DONE            02890000
*------------------------------------------------------------------*    02900000
* COMPARE THE MAJOR NAME,   R2=RNLE, R1=ENQPL                           02910000
*------------------------------------------------------------------*    02920000
*                                                                       02930000
         LA    R10,NQXP_OP_QNAME       QNAME ADDRESS                    02940000
*                                                                       02950000
         TM    RNLEFLGS,RNLEPATT       TYPE PATTERN                     02960000
         BO    TYPE_PA1                                                 02970000
*                                                                       02980000
         CLC   0(8,R10),RNLEQNME       CHECK    QNAME                   02990000
         BNE   NEXTRNL1                NO => GET NEXT RNL ENTRY         03000000
         B     CHK_MIN                                                  03010000
*                                                                       03020000
TYPE_PA1 EQU   *                                                        03030000
         LA    WORKREG,RNLEQNME        POINTER TO RNL QNAME             03040000
*                                                                       03050000
* EVALUATE MAJORs LENGTH I.E.QNAME=(P?P*) HAS LENGTH=4                  03060000
*                                                                       03070000
         LA    R8,8                    MAX LENGTH, RNLE                 03080000
         LA    RNAMELEN,0                                               03090000
CHK_LEN  EQU   *                                                        03100000
         CLI   0(WORKREG),X'40'                                         03110000
         BE    CHK_DONE                                                 03120000
         LA    RNAMELEN,1(0,RNAMELEN)  + 1 RNLE MAJOR LEN               03130000
         LA    WORKREG,1(0,WORKREG)    NEXT CHAR                        03130100
         BCT   R8,CHK_LEN                                               03130200
CHK_DONE EQU   *                                                        03130300
         LA    WORKREG,RNLEQNME        reset POINTER TO RNL QNAME       03130400
*                                                                       03130500
* now evaluate enq qname length, R10 pointer to qname                   03130600
*                                                                       03130700
         LA    R8,8                    MAX LENGTH, ENQ Major            03130800
         LA    R7,0                                                     03130900
CHK_LEN1 EQU   *                                                        03131000
         CLI   0(R10),X'40'                                             03131100
         BE    CHK_DON1                                                 03131200
         LA    R7,1(R7)                + 1 ENQ  MAJOR LEN               03131300
         LA    R10,1(0,R10)            NEXT CHAR                        03131400
         BCT   R8,CHK_LEN1                                              03131500
CHK_DON1 EQU   *                                                        03131600
         LA    R10,NQXP_OP_QNAME       reset QNAME POINTER              03131700
*                                                                       03131800
         BAL   R6,PAT_SCAN                                              03131900
         B     CHK_MIN     OFFSET --> 0   MATCH                         03132000
         B     NEXTRNL1    OFFSET --> 4   NO MATCH                      03132100
```

```
*                                                                       03132200
*-----------------------------------------------------------------*     03132300
* COMPARE THE MINOR NAME,   R2=RNLE, R1=ENQPL                           03132400
*-----------------------------------------------------------------*     03132500
CHK_MIN  EQU   *                                                        03132600
         SLR   RNAMELEN,RNAMELEN       CLEAR WORK REG                   03132700
         IC    RNAMELEN,RNLERNML       GET LENGTH OF RNAME IN THE RNLE  03132800
*                                                                       03132900
         LTR   RNAMELEN,RNAMELEN       MINOR LENGTH ZERO,               03133000
         BZ    MODEXIT1                RNL ENTRY WITHOUT MINOR =>MATCH  03134004
*                                                                       03135000
         ICM   R10,15,NQXP_OP_RNAME@   RNAME POINTER                    03136000
         SR    R7,R7                                                    03137000
         IC    R7,NQXP_OP_RNAMELEN        RNAME LEN                     03138000
*----------------------                                                 03139000
*---TEST RNL ENTRY TYPE                                                 03140000
*----------------------                                                 03150000
         TM    RNLEFLGS,RNLEGENR+RNLEPATT    TYPE SPECIFIC              03160000
         BNZ   CHK_TGEN                                                 03170000
         CR    R7,RNAMELEN             LENGTH SHOULD BE EQUAL           03180000
         BNE   NEXTRNL1                                                 03190000
         B     COMPARE                                                  03200000
CHK_TGEN EQU   *                                                        03210000
         TM    RNLEFLGS,RNLEGENR       GENERIC  TYPE                    03220000
         BZ    TYPE_PAT                                                 03230000
         CR    R7,RNAMELEN             ENQ RNAME LEN SHOULD BE HI OR EQ 03231000
         BL    NEXTRNL1                                                 03232000
         B     COMPARE                                                  03232100
*                                                                       03232200
TYPE_PAT EQU   *                                                        03232300
         LA    WORKREG,RNLERNME        POINTER TO RNL RNAME             03232400
         BAL   R6,PAT_SCAN                                              03232500
         B     MODEXIT     OFFSET --> 0   MATCH                         03232600
         B     NEXTRNL1    OFFSET --> 4   NO MATCH                      03232700
*--------------------------------------TYPE SPECIFIC & GENERIC          03232800
COMPARE  EQU   *                       R10=ENQ MINOR NAME               03232900
         BCTR  RNAMELEN,ZERO           ADJUST LENGTH                    03233000
         EX    RNAMELEN,COMPRNME       COMPARE THE RNAME                03233100
         BE    MODEXIT1                MATCH   => GO TO GRS RNL PROC    03233200
         B     NEXTRNL1                UNEQUAL => GET NEXT RNLE         03233300
*                                                                       03233400
*-----------------------------------------------------------------      03233500
* PATTERN SCAN         R7=ENQ NAME LENGTH, R10=ENQ Q/RNAME POINTER      03233600
*                      R5=RNL NAME LENGTH, R12=RNL Q/RNAME POINTER      03233700
*                      R6=RETURN ADDRESS  +0=MATCH, +4=NOMATCH          03233800
*                                                                       03233900
* LOGIC:                                                                03234000
*       RNLE is the source string,  Q/RNAME is the target string        03234100
*                                                                       03234200
*       compare strings one char at a time                              03234300
*         if source is char ? increment source and target               03234400
*         if source is char * increment target                          03234500
*         if match increment source and targett                         03234600
*         if no_match return 4=nomatch                                  03234700
*                                                                       03234800
*         when source or target string is fully scanned (length=0)      03234900
*                                                                       03235000
*         if source=0  & target=0   -->  match (return offset 0)        03235100
*         if source=0  & target>0 & last source char=* --> match        03235200
*         if source=0  & target>0   -->  no_mat(return offset 4)        03235300
*         if source>0  & target=0   -->  no_mat(return offset 4)        03235400
*                                                                       03235500
*-----------------------------------------------------------------      03235600
PAT_SCAN EQU   *                                                        03235700
         SR    R8,R8                      RNL RNAME INCREMENT REG       03235800
         SR    R9,R9                      ENQ RNAME INCREMENT REG       03235900
         LA    R8,0(R8,WORKREG)        INCREMENT RNL POINTER            03236000
         LA    R9,0(R9,R10)            INCREMENT RNAME POINTER          03237000
PATTERN  EQU   *                                                        03238000
         CLI   0(R8),C'?'              PATTERN CHAR                     03239000
         BE    INCRM_ST                +1 SOURCE & TARGET               03239100
         CLI   0(R8),C'*'              PATTERN CHAR                     03239200
         BE    INCRM_2                                                  03239300
         CLC   0(1,R8),0(R9)                                            03239400
         BE    INCRM_ST                + 1 SOURCE & TARGET              03239500
         LA    R6,4(0,R6)              RETURN NO MATCH                  03239600
         BR    R6                                                       03239700
INCRM_ST EQU   *                                                        03239800
         BCTR  RNAMELEN,ZERO           DECREMENT RNL RNAME LEN          03239900
         BCTR  R7,ZERO                 DECREMENT ENQ RNAME LEN          03240000
         LTR   RNAMELEN,RNAMELEN                                        03241000
         BZ    SOURCE_O                                                 03242000
         LTR   R7,R7                                                    03242100
         BZ    END_SRC                 source>0 & target=0 -> no_mat    03242200
*                                      source>0 & target>0              03242300
         LA    R8,1(0,R8)              INCREMENT RNL NAME POINTER       03242400
         LA    R9,1(0,R9)              INCREMENT ENQ MINOR NAME POINTER 03242500
         B     PATTERN                                                  03242600
SOURCE_O EQU   *                                                        03242700
         LTR   R7,R7                                                    03242800
         BZ    MATCH_S                 source=0 & target=0 -> match     03242900
         CLI   0(R8),C'*'              LAST SOURCE CHAR *               03243000
         BE    MATCH_S                 SOURCE=0 & LAST CHAR *->MATCH    03243100
         B     END_MIN                 source=0 & target>0 -> no_mat    03243200
*                                                                       03243300
INCRM_T  EQU   *                                                        03243400
         BCTR  R7,ZERO                DECREMENT ENQ RNAME LEN           03243500
         LTR   R7,R7                                                    03243600
         BZ    END_MIN                 source>0 & target=0  no_mat      03243700
         LA    R9,1(0,R9)              INCREMENT ENQ MINOR NAME POINTER 03243800
         B     PATTERN1                                                 03243900
*                                                                       03244000
INCRM_2  EQU   *                       RNL NAME                         03245000
         BCTR  RNAMELEN,ZERO           DECREMENT RNL RNAME LEN          03246000
         LTR   RNAMELEN,RNAMELEN                                        03246100
         BZ    MATCH_S                 source=0 last char * -> match    03246200
         BCTR  R7,ZERO                 DECREMENT ENQ RNAME LEN          03246300
         LTR   R7,R7                                                    03246400
         BZ    END_MIN                 source >0 & target=0 -> no_mat   03246500
         LA    R8,1(0,R8)              INCREMENT RNL NAME POINTER       03246600
         LA    R9,1(0,R9)              INCREMENT ENQ MINOR NAME POINTER 03246700
         B     PATTERN1                                                 03246800
PATTERN1 EQU   *                                                        03246900
*                                      EX (PIPPO.V*??) '*' used like'?' 03247000
         CLI   0(R8),C'?'              PATTERN CHAR                     03248000
         BE    INCRM_ST                +1 SOURCE & TARGET               03248100
*                                      EX (PIPPO.V**?) '*' used like'?' 03248200
         CLI   0(R8),C'*'              PATTERN CHAR                     03248300
         BE    INCRM_ST                +1 SOURCE & TARGET               03248400
         CLC   0(1,R8),0(R9)                                            03248500
         BNE   INCRM_T                 + 1 TARGET                       03248600
         B     INCRM_ST                + 1 SOURCE & TARGET              03248700
*                                                                       03248800
MATCH_S  EQU   *                       RNL ENTRY FULLY SCANNED          03248900
         LA    R6,0(0,R6)              RETURN MATCH                     03249000
         BR    R6                                                       03250000
*                                                                       03250100
END_SRC  EQU   *                       RNL RNAME FULLY SCANNED,         03250200
END_MIN  EQU   *                       ENQ RNAME FULLY SCANNED,         03250300
         LA    R6,4(0,R6)              RETURN NO MATCH                  03250400
         BR    R6                                                       03250500
*----------------------------------------------------------------       03250600
* END PATTERN SCAN                                                      03250700
*----------------------------------------------------------------       03250800
         SPACE 3                                                        03250900
*-----------------------------------------------------------------*     03251000
NOMATCH  EQU   *                        SEARCH CONVERSION LIST          03252000
*-----------------------------------------------------------------*     03253000
         ICM   RNLEPTR,15,GVTRCRNL      RESERVE CONVERTION RNL          03254000
         BZ    MODEXIT                  NO RNL CONVERSION               03255000
         L     R14,COMP1                RETURN ADDRESS                  03256000
*                                                                       03257000
***      USING RNLE,RNLEPTR                                             03258000
*-----------------------------------------------------------------*     03259000
* NQXP_RD_VOLSER = VOLSER OF REQUESTED UCB                              03260000
*-----------------------------------------------------------------*     03270000
         SPACE                                                          03280000
COMPRNL1 EQU   *                       COMPARE PEL NAME TO RNL ENTRY    03290000
         TM    RNLEFLGS,RNLELAST       AT END OF THE RNL ?              03300000
         BO    NOHARDW                 YES => RNL SCAN DONE             03310000
         SPACE                                                          03320000
* COMPARE QNAME HWRESERV   WITH RNLE                                    03330000
*                                                                       03340000
         TM    RNLEFLGS,RNLEPATT       PATTERN TYPE ENTRY NOT           03350000
         BO    NEXTRNL1                SUPPORTED                        03360000
*                                                                       03370000
         CLC   HRDWNAME,RNLEQNME       CHECK IF QNAME HWRESERV          03380000
         BNE   NEXTRNL1                NO => GET NEXT RNL ENTRY         03390000
         SPACE                                                          03400000
* COMPARE THE VOLSER THAT MUST BE SPECIFIED                             03410000
         SLR   RNAMELEN,RNAMELEN       CLEAR WORK REG                   03420000
         IC    RNAMELEN,RNLERNML       GET LENGTH OF RNAME IN THE RNLE  03430000
*                                                                       03440000
         LTR   RNAMELEN,RNAMELEN       MINOR LENGTH CANNOT BE ZERO,     03450000
         BZ    NEXTRNL1                VOLSER MUST BE INDICATED         03460000
*                                                                       03470000
         BCTR  RNAMELEN,ZERO           ADJUST LENGTH                    03480000
         EX    RNAMELEN,COMPRNM1       COMPARE THE RNAME (VOLSER)       03490000
         BNE   NEXTRNL1                UNEQUAL => GET NEXT RNLE         03500000
         SPACE                                                          03510000
*                                      VOLUME SERIAL FOUND IN THE RNL   03520000
         OI    NQXP_REQUESTFLAGS1,NQXP_RF1_BYPASSRNLS                   03530000
*                                      INDICATE BYPASS RNL PROCESSING   03540000
         B     MODEXIT                 EXIT PROCESSING COMPLETE         03550000
*                                                                       03560000
* GET THE ADDRESS OF THE NEXT RNL ENTRY                                 03570000
*                                                                       03580000
NEXTRNL1 EQU   *                                                        03590000
         LA    FLENRNLE,RNLERNME-RNLE  LENGTH OF FIXED PART OF RNLE     03600000
         SLR   WORKREG,WORKREG         CLEAR WORK REG                   03610000
         IC    WORKREG,RNLERNML        GET RNAME LENGTH (VARIABLE)      03620000
         ALR   WORKREG,FLENRNLE        ADD FIXED + VARIABLE LENGTHS     03630000
         ALR   RNLEPTR,WORKREG         GET ADDRESS OF NEXT RNL ENTRY    03640000
         BR    R14                     CHECK THE NEW RNL ENTRY          03650000
*                                                                       03660000
COMPRNM1 CLC   RNLERNME(ZERO),NQXP_RD_VOLSER    COMPARE VOLSER          03670000
*OMPRNM1 CLC   RNLERNME(ZERO),UCBVOLI-UCBOB(R6) COMPARE VOLSER          03680000
*                                                                       03690000
COMPRNME CLC   RNLERNME(ZERO),0(R10)            COMPARE MINOR  **       03700000
*                                                                       03710000
COMP1    DC   A(X'80000000'+COMPRNL1)                                   03720000
COMPE    DC   A(X'80000000'+COMPRNLE)                                   03730000
*                                                                       03740000
         DROP  R3                                                       03750000
         DROP  ENQPL                                                    03760000
         DROP  RNLEPTR                                                  03770000
*--------------------------------------------------------------------   03780000
* R13 SAVE AREA, R14, RETURN ADDRESS, R15 RET-CODE                      03790000
*--------------------------------------------------------------------   03800000
MODEXIT1 EQU   *                                                        03820000
NOHARDW  EQU   *                                                        03821004
MODEXIT  EQU   *                                                        03830000
         SR    REG15,REG15             RETURN CODE 0                    03840000
         L     R14,12(0,REG13)         RECOVER THE RETURN ADDRESS       03850000
         LM    R0,R12,20(R13)          RECOVER OTHERS EXCEPT R15        03860000
         BSM   0,R14                   RETURN TO THE CALLER             03870000
*                                                                       03880000
*--------------------------------------------------------------------*  03890000
* DATA USED                                                           * 03900000
*--------------------------------------------------------------------*  03910000
         DS    0H                                                       03920000
HRDWNAME DC    CL8'HWRESERV'           MAJOR NAME FOR ALWAYS RESERVE    03930000
*--------------------------------------------------------------------*  03940000
         DROP  BASEREG                                                  03950000
         EJECT                                                          03960000
         ISGRNLE                                                        03970000
         PRINT NOGEN                                                    03980000
         ISGYNQXP                                                       03990000
         ISGGVT                                                         04000000
         CVT  DSECT=YES                                                 04010000
         IHAPSA                                                         04020000
         IEFUCBOB DEVCLAS=DA                                            04030000
*                                                                       04040000
         END                                                            04050000
//LINK  EXEC  PGM=IEWL,PARM='LIST,RENT,XREF'                            04060000
//SYSUT1 DD SPACE=(CYL,(1,1)),UNIT=SYSDA                                04070000
//SYSLMOD DD DSN=SYS1.SANDBOX.LINKLIB,DISP=SHR                          04080000
//SYSPRINT DD SYSOUT=*                                                  04090000
//SYSPUNCH  DD DSN=&OBJ(AUDIT),DISP=(OLD,PASS)                          04100000
//SYSUDUMP  DD SYSOUT=*                                                 04110000
//SYSLIN DD *                                                           04120000
 INCLUDE SYSPUNCH(AUDIT)                                                04130000
 ENTRY ISGCVXIT                                                         04140000
 NAME ISGNQXIT(R)                                                       04150000
```

Related publications

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

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 310.

- *z/OS Infoprint Server Implementation*, SG24-6234
- *Open Source Software for z/OS and OS/390 UNIX*, SG24-5944

Other resources

These publications are also relevant as further information sources:

- *z/OS Planning for Installation*, GA22-7504
- *z/OS MVS Programming: Assembler Services Reference*, SA22-7606
- *z/OS MVS Planning: Global Resource Serialization*, SA22-7600
- *z/OS MVS Installation Exits*, SA22-7593
- *z/OS UNIX System Services Planning*, GA22-7800
- *z/OS UNIX System Services Command Reference*, SA22-7802
- *z/OS Distributed File Service DFS Administration*, SC24-5915
- *z/OS Distributed File Service Messages and Codes*, SC24-5917
- *z/OS Security Server RACF Security Administrator’s Guide*, SA22-7683
- *z/OS MVS Interactive Problem Control System (IPCS) User's Guide*, SA22-7596
- *z/OS MVS Interactive Problem Control System (IPCS) Commands*, SA22-7594
- *z/OS MVS Diagnosis: Tools and Service Aids*, GA22-7589
- *z/OS Security Server RACF Command Language Reference*, SA22-7687
- *z/OS Security Server RACF Diagnosis Guide*, GA22-7689
- *z/OS Security Server RACF Data Areas*, GA22-7680
- *z/OS Security Server RACF Callable Services*, SA22-7691
- *z/OS Security Server RACF Command Language Reference*, SA22-7687
- *z/OS Security Server RACF Messages and Codes*, SA22-7686
- *z/OS Security Server RACF Security System Programmer’s Guide*, SA22-7681
- *z/OS SecureWay Security Server Firewall Technologies*, SC24-5922
- *z/OS SecureWay Security Server Network Authentication Service Administration*, SC24-5926
- *z/OS SecureWay Security Server Network Authentication Service Programming*, SC24-5927
- *DB2 Universal Database for OS/390 and z/OS Version 7 Administration Guide*, SC26-9931

- *z/OS: Communications Server: IP Configuration Reference*, SC31-8776
- *z/OS: Communications Server: IP Configuration Guide*, SC31-8775.
- *z/OS Integrated Cryptographic Service Facility System Programmer's Guide*, SA22-7520
- *z/OS Integrated Cryptographic Service Facility Administrator‘s Guide*, SA22-7521
- *z/OS Integrated Cryptographic Service Application Programmer's Guide*, SA22-7522

Referenced Web sites

These Web sites are also relevant as further information sources:

- z/OS UNIX System Services home page:

  `http://www.ibm.com/servers/eserver/zseries/zos/unix`

- A Goal Mode Migration Aid tool to help get to goal mode can be downloaded from:

  `http://www.ibm.com/servers/eserver/zseries/zos/rmf/rmfhtmls/rmftools.htm#spr_goal`

How to get IBM Redbooks

Search for additional Redbooks or Redpieces, view, download, or order hardcopy from the Redbooks Web site:

ibm.com/redbooks

Also download additional materials (code samples or diskette/CD-ROM images) from this Redbooks site.

Redpieces are Redbooks in progress; not all Redbooks become Redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

IBM Redbooks collections

Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.

Index

z/OS Version 1 Release 2 Implementation

(0.5” spine)
0.475”<->0.873”
250 <-> 459 pages

z/OS Version 1 Release 2 Implementation

64-bit virtual storage addressing, GRS, SMP/E, WLM, RMF, LE, Kerberos

zSeries File System, JES2, JES3, SDSF, UNIX System Services

RACF, Firewall technology, Cryptographic Services

This IBM Redbook highlights many new enhancements made in z/OS Version 1 Release 2. You can use this information to help you install, tailor, and configure z/OS Version 1 Release 2.

It will give you an understanding of a new architecture for 64-bit virtual storage addressing. Other topics discussed are the enhancements to z/OS base control program (BCP) elements; z/OS JES2 Version 1 Release 2, a new zSeries File System, Spool Display and Search Facility (SDSF), UNIX System Services, SMP/E, Workload Manager, RMF, z/OS JES3 Version 1 Release 2, and Language Environment.

This redbook also describes several SecureWay Security Server functional enhancements to RACF, Firewall technology, z/OS cryptographic services, and Security Server Kerberos.

This Redbook is intended for system programmers and administrators responsible for customizing and installing z/OS.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

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

For more information:
ibm.com/redbooks

SG24-6235-00 ISBN 0738422924