RACF Version 2 Release 1
Installation and Implementation Guide

Document Number GG24-4405-00

December 1994

International Technical Support Organization
Poughkeepsie Center

Take Note!

Before using this information and the product it supports, be sure to read the general information under “Special Notices” on page xiii.

First Edition (December 1994)

This edition applies to Version 2 Release 1 of Resource Access Control Facility (RACF), Program Number 5695-039 for use with MVS/ESA Version 4 Release 2.2 or later.

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

An ITSO Technical Bulletin Evaluation Form for reader's feedback appears facing Chapter 1. If the form has been removed, comments may be addressed to:

IBM Corporation, International Technical Support Organization
Dept. H52 Mail Station P099
522 South Road
Poughkeepsie, New York 12601-5400

Abstract

This document describes the installation and implementation of Resource Access Control Facility Version 2 Release 1.

This document is intended for system programmers, security administrators, and those who need to plan for and install RACF 2.1.0. This document focuses on the implementation of RACF 2.1.0 in a sysplex environment.

It contains recommendations and examples of how to use and implement the functions as well as a short description of how they work. Sample programs are provided to illustrate the use of some of the new RACF features.

(116 pages)

Contents

Figures

Tables

Special Notices

This publication is intended to help system programmers and RACF security administrators in planning, installing and customizing Resource Access Control Facility Version 2 Release 1 in a MVS/ESA environment. The information in this publication is not intended as the specification of any programming interfaces that are provided by RACF 2.1.0. See the PUBLICATIONS section of the IBM Programming Announcement for RACF 2.1.0 for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter 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 the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

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

AIX	CBIPO	CBPDO
CICS	CICS/ESA	CustomPac
DATABASE 2	DB2	DFSMS
DFSMSdfp	DFSMSdss	DFSMShsm
ES/9000	FunctionPac	Hardware Configuration Definition
Hiperbatch	IBM	Library Reader
MVS/DFP	MVS/ESA	MVS/SP
MVS/XA	OPC	OpenEdition
ProductPac	ProductPac/E	PROFS
QMF	RACF	S/390
SystemPac	SystemView	

The following terms are trademarks of other companies:

POSIX	Insitute of Electrical and Electronics Engineers
UNIX	UNIX System Laboratories, Inc.

Other trademarks are trademarks of their respective companies.

Preface

This publication is intended primarily for customer personnel responsible for planning, installing and customizing Resource Access Control Facility Version 2 Release 1 in a MVS/ESA environment. Readers would include IBM MVS/ESA and large system product specialists, systems programmers, and RACF security administrators. This document focus on the sysplex aspects of the installation and customization tasks.

How This Document is Organized

The document is organized as follows:

- Chapter 1, “Introduction to RACF Version 2 Release 1”

  This chapter provides an introduction to the new functions of RACF 2.1.0.

- Chapter 2, “RACF Version 2 Release 1 Installation”

  This chapter provides an overview of the installation steps for RACF 2.1.0.

- Chapter 3, “Sysplex Communication and Data Sharing”

  This chapter describes how to implement the RACF sysplex communication and RACF sysplex data sharing.

- Chapter 4, “RACF Dynamic Exits”

  This chapter provides an introduction to the MVS/ESA Dynamic Exit facility. It also describes how this facility is used in a RACF environment.

- Chapter 5, “RACLIST Enhancements”

  This chapter provides an overview of the RACLIST enhancements. It describes also how this facility is implemented and tested in a sysplex environment.

- Chapter 6, “RACF OpenEdition MVS Support”

  This chapter describes how to implement security in a OpenEdition MVS environment.

- Appendix A, “Started Procedure Tables Report”

  This appendix provides information on the *started procedure tables report* that is produced by the RACF data security monitor (DSMON).

- Appendix B, “Convert ICHRIN03 to STARTED Class Profiles”

  This appendix describes how to convert entries from the *started procedures table* (ICHRIN03) into resource definition commands for the RACF class STARTED.

- Appendix C, “Database Range Table - Sample JCL”

  This appendix provides SMP/E information for the *RACF database range table*.

- Appendix D, “Database Name Table - Sample JCL”

  This appendix provides SMP/E information for the *RACF database name table*.

- Appendix E, “Started Procedures Table - Sample JCL”

  This appendix provides SMP/E information for the *started procedures table*.

- Appendix F, “Administrative Data Utility - Sample JCL”

  This appendix provides a example of the *administrative data utility* on how to load the various cache structures in the coupling facility.

- Appendix G, “Dynamic RACF Exit - Sample Code”

  This appendix provides the sample Assembler code on how to invoke a dynamic RACF exit call. This code is implemented in a *dynamic exit stub* that is replacing the code in a standard RACF exit.

- Appendix H, “Source Code for a Demonstration Program”

  This appendix provides the sample code for the demonstration program that is used to get a better understanding of the RACF sysplex communication and the new RACROUTE REQUEST=LIST,GLOBAL=YES enhancements.

Related Publications

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

A document whose order number begins with the prefix LY is available to IBM-licensed customers only.

RACF 2.1 Publications

The publications in Table 1 contain detailed information about RACF Version 2.

> In response to customer requirements related to documentation, information specific to operating RACF in a VM environment is not included in the RACF 2.1 library. Refer to RACF 1.9 and 1.9.2 publications if you need specific information about using RACF on VM systems. Documentation describing how to use RACF 2.1 in a shared database environment with MVS and VM systems remains in RACF 2.1 publications.

Table 1 (Page 1 of 2). The RACF Library

Task	Title	Order Number	Contents
Evaluation Planning	*RACF General Information*	GC23-3723	Contains an overview of the product and highlights the new functions for the current release
Planning	*RACF Publications Ordering Guide*	SX23-0461	Lists the order numbers of RACF publications
Planning Installation Customization Diagnosis	*RACF System Programmer's Guide*	SC23-3725	Describes how to modify and maintain RACF
Installation Customization Diagnosis	*RACF Macros and Interfaces*	SC23-3732	Describes each product macro and its syntax and explains how to code the interfaces

Table 1 (Page 2 of 2). The RACF Library

Task	Title	Order Number	Contents
Customization	*RACF External Security Interface (RACROUTE) Macro Reference*	GC23-3733	Describes the RACF system macros and explains how to code the interfaces
Customization	*RACF Callable Services*	GC23-3737	Describes the RACF callable services and explains how to code the interfaces
Planning Customization Administration	*RACF Security Administrator's Guide*	SC23-3726	Explains RACF concepts and describes how to plan for and implement RACF
Diagnosis	*RACF Diagnosis Guide*	LY27-2635	Explains how to diagnose problems in the RACF program product
Installation Customization Administration	*RACF Command Language Reference*	SC23-3731	Contains the functions and syntax of all RACF commands
Installation Customization Administration	*RACF Command Syntax Booklet*	SX23-0462	Contains information extracted from *RACF Command Language Reference*
Administration Diagnosis	*RACF Messages and Codes*	SC23-3730	Contains the RACF messages, routing and descriptor codes, RACF manager return codes, and RACF-related system completion codes
Planning Customization Administration	*RACF Auditor's Guide*	SC23-3727	Describes auditing considerations and how to use the RACF report writer and the data security monitor
End Use	*RACF General User's Guide*	SC23-3728	Explains how to perform common end-user tasks
Evaluation Planning Installation Customization Administration End Use Diagnosis	*RACF Master Index*	GC23-3724	Provides book-level references to subjects within the RACF library
Installation	*RACF Program Directory*	Shipped with the product	Describes how to install RACF
Diagnosis	*RACF Data Areas*	LY27-2636	Contains descriptions of data areas used by RACF
Planning Migration Installation Customization Administration Auditing Operation Application Development	*RACF Planning: Installation and Migration*	GC23-3736	Contains information to guide installations through the migration process from prior releases of RACF to RACF 2.1.0

IBM publishes a bibliography of security-related books: *Systems Security Publications Bibliography,* G320-9279.

Softcopy Publications

The RACF 2.1 library, with the exception of the licensed publications, is available on CD-ROM as part of the *Online Library Omnibus Edition MVS Collection Kit,* SK2T-0710. This softcopy collection contains a set of key MVS and MVS-related product books and includes the IBM Library Reader, a program that enables customers to read the softcopy books. It includes both the RACF 1.9.2 and RACF 2.1 libraries. *RACF Messages and Codes* is also available as part of *Online Library Productivity Edition Messages and Codes Collection,* SK2T-2068.

The *RACF Security CD-ROM*, SK2T-2180, contains the RACF 1.9, RACF 1.9.2, and RACF 2.1 product libraries (with the exception of the licensed publications) and product books from the MVS and VM collections, ITSO redbooks, and WSC orange books that contain substantial amounts of information related to RACF. Using this CD-ROM, you have access to RACF-related information from IBM products such as MVS, VM, CICS, and NetView without maintaining shelves of hardcopy documentation or handling multiple CD-ROMs.

The Previous Edition of RACF Publications

RACF 1.9.2 publications are available using the following order numbers:

- *RACF General Information*, GC28-0722
- *RACF Licensed Program Specifications*, GC28-0732
- *RACF Publications Ordering Guide*, GX22-0002
- *RACF System Programming Library*, SC28-1343
- *RACF Macros and Interfaces*, SC28-1345
- *External Security Interface (RACROUTE) Macro Reference for MVS and VM*, GC28-1366
- *RACF Security Administrator's Guide*, SC28-1340
- *RACF Diagnosis Guide*, LY28-1016
- *RACF Command Language Reference*, SC28-0733
- *RACF Command Syntax Booklet*, SX22-0014
- *RACF Messages and Codes*, SC38-1014
- *RACF Auditor's Guide*, SC28-1342
- *RACF General User's Guide*, SC28-1341
- *RACF Master Index*, GC28-1035
- *RACF Program Directory for MVS Installations*, GC28-1054
- *RACF Program Directory for VM Installations*, GC28-1034
- *RACF Data Areas*, LY28-1830
- *RACF Migration and Planning*, GC23-3054

Other Product Publications

Table 2 (Page 1 of 3). Security Publications for Other IBM Products		
Product	**Type of Information**	**Publications**
CICS/ESA Version 3	**Primary**	*CICS-RACF Security Guide*, SC33-0749
CICS/ESA Version 3	Related	*CICS/ESA:* • *System Definition Guide*, SC33-0664 • *Intercommunication Guide*, SC33-0657
CICS/ESA Version 4	**Primary**	*CICS-RACF Security Guide*, SC33-1185
DB2 Version 2	**Primary**	*IBM DATABASE 2 Version 2 Administration Guide*, SC26-4374
DB2 Version 2	Related	*IBM DATABASE 2 Version 2:* • *SQL Reference*, SC26-4380 • *General Information*, GC26-4373
DB2 Version 3	**Primary**	*IBM DATABASE 2 Version 3 Administration Guide*, SC26-4888
DB2 Version 3	Related	*IBM DATABASE 2 Version 3:* • *SQL Reference*, SC26-4890 • *General Information*, GC26-4886
DFDSS 2.5	**Primary**	For DFDSS-authorized storage administration: *Data Facility Data Set Services Version 2 Release 5:* • *Update Guide*, SC35-0146
DFDSS 2.5	Related	*Data Facility Data Set Services Version 2 Release 5:* • *User's Guide*, SC26-4388 • *Reference*, SC26-4389
DFHSM 2.6	Related	*Data Facility Hierarchical Storage Manager Version 2 Release 6:* • *Installation and Customization Guide*, SH35-0084 • *System Programmer's Guide*, SH35-0085 • *User's Guide*, SH35-0093
DFP	Related	See MVS/DFP
DFSMS/MVS 1.1	**Primary**	*MVS/ESA Storage Management Library: Managing Data Sets*, SC26-3124 *DFSMS/MVS Version 1 Release 1:* • *Managing Catalogs*, SC26-4914 • *Using Data Sets*, SC26-4922 • *Using Magnetic Tapes*, SC26-4923 • *Storage Administration Reference for DFSMSdfp*, SC26-4920 • *Storage Administration Guide for DFSMShsm*, SH21-1076 • *Storage Administration Reference for DFSMShsm*, SH21-1075 • *Storage Administration Guide for DFSMSdss*, SC26-4930 • *Storage Administration Reference for DFSMSdss*, SC26-4929 • *Access Method Services for the Integrated Catalog Facility*, SC26-4906 • *Access Method Services for VSAM Catalogs*, SC26-4905 *Data Facility Data Set Services Version 2 Release 5: Update Guide*, SC35-0146

Table 2 (Page 2 of 3). Security Publications for Other IBM Products		
Product	**Type of Information**	**Publications**
ICSF/MVS	**Primary**	*Integrated Cryptographic Service Facility/MVS Administrator's Guide*, SC23-0097
IMS 2.2	Related	*Information Management System: System Administration Guide*, SC26-4176
IMS/ESA Version 3	Related	*Information Management System/ESA Version 3: System Administration Guide*, SC26-4282
IMS/ESA Version 4	Related	*Information Management System/ESA Version 4: System Administration Guide*, SC26-3075
JES2 Version 4	Related	*MVS/ESA:* • *JES2 Initialization and Tuning Guide*, SC23-0082 • *JES2 Initialization and Tuning Reference*, SC23-0083 • *JES2 Customization*, LY28-1029
JES2 Version 5	Related	*MVS/ESA:* • *JES2 Initialization and Tuning Guide*, SC28-1453 • *JES2 Initialization and Tuning Reference*, SC28-1454 • *JES2 Macros*, SC28-1462 • *JES2 Installation Exits*, SC28-1463
JES3 Version 4	Related	*MVS/ESA:* • *JES3 Initialization and Tuning Guide*, SC23-0088 • *JES3 Initialization and Tuning Reference*, SC23-0089 • *JES3 Customization*, LY28-1026
JES3 Version 5	Related	*MVS/ESA:* • *JES3 Initialization and Tuning Guide*, SC28-1455 • *JES3 Initialization and Tuning Reference*, SC28-1456 • *JES3 Customization*, LY28-1853
LFS/ESA	**Primary**	*LAN File Services/ESA (LFS/ESA): MVS Guide and Reference*, SH24-5265
MQM MVS/ESA	**Primary**	*Message Queue Manager MVS/ESA: System Management Guide*, SC33-0806
MVS Version 4 BCP	Related	See Table 3 on page xxi.
MVS Version 5 BCP	Related	See Table 4 on page xxii.
MVS/DFP 3.3	Related	*MVS/ESA Storage Management Library: Managing Data Sets*, SC26-3124 *MVS/Data Facility Product (MVS/DFP) Version 3 Release 3:* • *Access Method Services for the Integrated Catalog Facility*, SC26-4562 • *Access Method Services for VSAM Catalogs*, SC26-4570 • *Interactive Storage Management Facility User's Guide*, SC26-4563 • *Managing Catalogs*, SC26-4555 • *Storage Administration Reference*, SC26-4566 • *Using Data Sets*, SC26-4749 • *Using Magnetic Tape Labels and File Structure*, SC26-4565
MVS/ESA OpenEdition	**Primary**	• *MVS/ESA OpenEdition: MVS Shell and Utilities Feature Program Directory*, GC23-3022 • *MVS/ESA Planning: OpenEdition MVS*, SC23-3015

Table 2 (Page 3 of 3). Security Publications for Other IBM Products		
Product	**Type of Information**	**Publications**
NetView Version 2 Release 3	**Primary**	*NetView 2.3:* • *Installation and Administration Guide (MVS),* SC31-6125
NetView/Access Services	**Primary**	*NetView/Access:* • *Installing and Operating,* SH12-5195 • *Administration and Messages,* SH12-5194
OPC/A	**Primary**	*Operations Planning and Control/Advanced: Installation and Customization,* SH19-6445
PSF/MVS Version 2	**Primary**	• *Print Services Facility Security Guide,* S544-3291 • *PSF/MVS: System Programming Guide,* S544-3672
SDSF 1.4	**Primary**	*System Display and Search Facility: Guide and Reference,* SC23-0408
SDSF 1.4	Related	*SDSF Customization and Security,* SC23-3807
TSO/E Version 2	Related	*TSO/E Version 2:* • *Customization,* SC28-1872 • *Programming Services,* SC28-1875 • *System Programming Command Reference,* SC28-1878 • *User's Guide,* SC28-1880
TSO/E ICF	Related	*TSO/E Information Center Facility: Administrator's Guide,* GC28-1332
VTAM 3.3	**Primary**	*Virtual Telecommunications Access Method (VTAM): Network Implementation Guide,* SC31-6404
VTAM 3.2 and 3.3	Related	*Virtual Telecommunications Access Method (VTAM):* • *Programming,* SC31-6409 • *Programming for LU 6.2,* SC31-6410
VTAM 3.4	Related	*Virtual Telecommunications Access Method (VTAM):* • *Programming,* SC31-6436 • *Programming for LU 6.2,* SC31-6437
VTAM 4.1	Related	*Virtual Telecommunications Access Method (VTAM):* • *Programming,* SC31-6421 • *Programming for LU 6.2,* SC31-6425
VTAM 4.2	Related	*Virtual Telecommunications Access Method (VTAM):* • *Programming,* SC31-6496 • *Programming for LU 6.2,* SC31-6497

Related Publications for MVS Version 4

Table 3 lists the publications in the MVS Version 4 library to which this book refers. For each publication, the table gives the short title that is used in the text of this book, the publication's full title, and its order number.

Table 3 (Page 1 of 2). Related MVS Version 4 Publications	
Short Title	**Full Title and Order Number**
MVS Authorized Assembler	*MVS/ESA Application Development Guide: Authorized Assembler Language Programs,* GC28-1645
MVS Configuration Program	*MVS/ESA: MVS Configuration Program,* GC28-1615

Table 3 (Page 2 of 2). Related MVS Version 4 Publications	
Short Title	**Full Title and Order Number**
MVS Hiperbatch	*MVS/ESA Application Development Guide: Hiperbatch*, GC28-1673
MVS Initialization and Tuning Guide	*MVS/ESA: Initialization and Tuning Guide*, GC28-1634
MVS Initialization and Tuning Reference	*MVS/ESA: Initialization and Tuning Reference*, GC28-1635
MVS JCL	*MVS/ESA: JCL User's Guide*, GC28-1653 *MVS/ESA: JCL Reference*, GC28-1654
MVS Planning: APPC Management	*MVS/ESA Planning: APPC Management*, GC28-1110
MVS Planning: B1 Security	*MVS/ESA Planning: B1 Security*, GC28-1171
MVS Planning: Operations	*MVS/ESA Planning: Operations*, GC28-1625
MVS Planning: Problem Determination and Recovery	*MVS/ESA Planning: Problem Determination and Recovery*, GC28-1629
MVS Planning: Security	*MVS/ESA Planning: Security*, GC28-1604
MVS Planning: Sysplex Management	*MVS/ESA Planning: Sysplex Management*, GC28-1620
MVS SMF	*MVS/ESA: System Management Facilities (SMF)*, GC28-1628
MVS System Commands	*MVS/ESA: System Commands*, GC28-1626

Related Publications for MVS Version 5

Table 4 lists the publications in the MVS Version 5 library to which this book refers. For each publication, the table gives the short title that is used in the text of this book, the publication's full title, and its order number.

Table 4 (Page 1 of 2). Related MVS Version 5 Publications	
Short Title	**Full Title and Order Number**
MVS Authorized Assembler	*MVS/ESA Programming: Authorized Assembler Services Guide*, GC28-1467
MVS HCD User's Guide	*MVS/ESA Hardware Configuration Definition: User's Guide*, SC33-6468
MVS Hiperbatch	*MVS/ESA Programming: Hiperbatch Guide*, GC28-1470
MVS Initialization and Tuning Guide	*MVS/ESA: Initialization and Tuning Guide*, SC28-1451
MVS Initialization and Tuning Reference	*MVS/ESA: Initialization and Tuning Reference*, SC28-1452
MVS JCL	*MVS/ESA: JCL User's Guide*, GC28-1473 *MVS/ESA: JCL Reference*, GC28-1479
MVS Planning: APPC Management	*MVS/ESA Planning: APPC Management*, GC28-1503
MVS Planning: B1 Security	*MVS/ESA Planning: B1 Security*, GC28-1440
MVS Planning: Operations	*MVS/ESA Planning: Operations*, GC28-1441
MVS Planning: Problem Determination and Recovery	*MVS/ESA Diagnosis: Procedures*, LY28-1844
MVS Planning: Security	*MVS/ESA Planning: Security*, GC28-1439
MVS Setting Up a Sysplex	*MVS/ESA Setting Up a Sysplex*, GC28-1449

Table 4 (Page 2 of 2). Related MVS Version 5 Publications	
Short Title	**Full Title and Order Number**
MVS SMF	*MVS/ESA: System Management Facilities (SMF),* GC28-1457
MVS System Commands	*MVS/ESA: System Commands,* GC28-1442

System/390 MVS Sysplex Library Publications

Table 5 lists the publications in the System/390 MVS Sysplex library to which this book refers. For each publication, the table gives the short title that is used in the text of this book, the publication's full title, and its order number.

Table 5. Related Sysplex Library Publications	
Short Title	**Full Title and Order Number**
Sysplex Overview	*System/390 MVS Sysplex Overview*, GC28-1208
Sysplex Systems Management	*System/390 MVS Sysplex Systems Management*, GC28-1209
Sysplex Hardware and Software Migration	*System/390 MVS Sysplex Hardware and Software Migration*, GC28-1210
Sysplex Application Migration	*System/390 MVS Sysplex Application Migration*, GC28-1211

MVS/ESA OpenEdition Library Publications

Table 6 lists the publications in the MVS/ESA OpenEdition library to which this book refers. For each publication, the table gives the short title that is used in the text of this book, the publication's full title, and its order number.

Table 6. MVS/ESA OpenEdition Library Publications	
Short Title	**Full Title and Order Number**
Introducing OpenEdition MVS	*Introducing OpenEdition MVS*, GC23-3010
OpenEdition MVS User's Guide	*MVS/ESA OpenEdition MVS User's Guide*, SC23-3013
OpenEdition MVS Command Reference	*MVS/ESA OpenEdition MVS Command Reference*, SC23-3014
Planning: OpenEdition MVS	*MVS/ESA Planning: OpenEdition MVS*, SC23-3015
Assembler Callable Services for OpenEdition MVS	*MVS/ESA Application Development Reference: Assembler Callable Services for OpenEdition MVS*, SC23-3020

IBM Systems Center Publications

IBM Systems Centers produce "red" and "orange" books that can be helpful in setting up and using RACF.

These books have not been subjected to any formal review nor have they been checked for technical accuracy, but they represent current product understanding (at the time of their publication) and provide valuable information on a wide range of RACF topics. They are not shipped with RACF. You must order them separately.

For your convenience, the following is a selected list of these books:

- *APPC/MVS Technical Presentation Guide,* GG24-3848
- *DFSMS and RACF Usage Considerations,* GG24-3378
- *MVS 3.1.3 and RACF 1.9 Security Implementation Guide,* GG24-3585
- *RACF Starter System for MVS,* GG24-3120
- *Tutorial: Options for Tuning RACF,* GG22-9396
- *MVS/ESA Security Enhancement Presentation Foils,* GG24-3470
- *Introduction to System and Network Security: Considerations, Options and Techniques,* GG24-3451
- *MVS Software Management Cookbook,* GG24-3481
- *MVS/ESA OpenEdition DCE: Application Development Cookbook,* GG24-4481
- *MVS/ESA OpenEdition DCE: Application Support Server for IMS and CICS,* GG24-4482
- *MVS/ESA OpenEdition DCE: Presentation Guide Volume 1,* GG24-4240
- *MVS/ESA Support for IEEE Posix Standards: Technical Presentation Guide,* GG24-3867
- *OpenEdition MVS for MVS/ESA 5.1 Presentation Guide,* GG24-4095
- *DFSMS/MVS Version 1 Release 2.0 Support for OpenEdition MVS,* GG24-4401
- *TCP/IP: Tutorial and Technical Overview,* GG24-3376
- *System/390 MVS Sysplex Hardware and Software Migration,* GC28-1210
- *Elements of Security: RACF Overview - Student Notes,* GG24-3970
- *Elements of Security: RACF Installation - Student Notes,* GG24-3971
- *Elements of Security: RACF Advanced Topics - Student Notes,* GG24-3972
- *RACF Version 2 Release 1 Presentation Guide,* GG24-4281
- *Enhanced Exploitation of RACF 1.9.2 Secured Single Signon Using NV/AS,* GG24-4184
- *Enhanced Exploitation of RACF 1.9.2 SDSF Conversion,* GG24-4085
- *RACF Macros and Exit Coding,* GG24-3984
- *Enhanced Auditing Using the RACF SMF Data Unload Utility*, GG24-4453
- *Secured Single Signon in a Client/Server Environment*, GG24-4282

Other books are available, but they are not included in this list either because the information they present has been incorporated into IBM product manuals or because their technical content is outdated.

A complete list of International Technical Support Organization publications, with a brief description of each, may be found in:

> *International Technical Support Organization Bibliography of Redbooks* GG24-3070.

To get listings of ITSO technical bulletins (redbooks) online, VNET users may type:

```
TOOLS SENDTO WTSCPOK TOOLS REDBOOKS GET REDBOOKS CATALOG
```

How to Order ITSO Technical Bulletins (Redbooks)

IBM employees in the USA may order ITSO books and CD-ROMs using PUBORDER. Customers in the USA may order by calling 1-800-879-2755 or by faxing 1-800-284-4721. Visa and Master Cards are accepted. Outside the USA, customers should contact their IBM branch office.

Customers may order hardcopy redbooks individually or in customized sets, called GBOFs, which relate to specific functions of interest. IBM employees and customers may also order redbooks in online format on CD-ROM collections, which contain the redbooks for multiple products.

Acknowledgments

This publication is the result of a residency conducted at the ITSO in Poughkeepsie N.Y.

We would like to acknowledge the excellent work done by the following people:

Theo Jenniskens
IBM Germany

Horst Martin Dzatkowski
IBM Germany

Peter Hilger
IBM Germany

Erik Ankjer Pauner
IBM Denmark

Susan Thompson
IBM Kingston

During the preparation of the material in this document, many people in IBM were of help. Of these, Walt Farrell, Dionne Graff, Debbie Mapes and Mark Nelson of RACF Design should be mentioned for their assistance in getting all the details right.

Cees Kingma
International Technical Support Organization, Poughkeepsie

Chapter 1. Introduction to RACF Version 2 Release 1

This chapter describes the new functions in RACF 2.1.0 running on MVS. The intent of this introduction is to give the reader an idea of the concepts and facilities of some of the new function, and their benefits in a customer environment.

RACF 2.1.0 contains two major functions and some smaller new functions. The major functions are:

- OpenEdition MVS support, which provides security for the new OpenEdition Service Features of MVS/ESA Version 4 Release 3 and later releases.
- RACF sysplex data sharing and RACF sysplex communication, which provides improvements in system performance, system management, and system availability for customers running MVS/ESA Version 5 Release 1 and using a coupling facility.

In addition to the major line items, there are several end-user and system management enhancements in RACF 2.1.0. Among these enhancements there will be *dynamic started task table* support and a new tool that allows the customers to upload the RACF SMF records to a relational database management system of an installation's choice. The installation can extract information from this database by using SQL statements or QMF queries.

For information on RACF 2.1.0, refer to *RACF V2.1 Presentation Guide*. For more detailed information on how to exploit the new RACF SMF Data Unload utility, refer to *Enhanced Auditing Using the RACF SMF Data Unload Utility*.

This chapter will also provide a brief look at the new functions and enhancements made in MVS/ESA sysplex environment. This document will not cover in great detail the MVS/ESA SP 5.1 and MVS/ESA SP 5.2 related enhancements. For details on these topics, please read the publication sections of this redbook to find the appropriate documentation.

1.1 Enhanced Functions in a Sysplex

MVS/ESA SP Version 5 Release 1 expands the concept of sysplex introduced with MVS/ESA SP Version 4. The updates to MVS/ESA in Version 5 Release 1 focus on the following areas:

- Increased availability of customer data
- Application availability improvements
- System management improvements
- Workload management simplification

The new cross-system extended services (XES) and enhanced event notification facility increase the availability of data in MVS/ESA SP 5.1. Cross-system extended services provides a new set of system services to provide communication to a *coupling facility* allowing:

- High performance data sharing across the sysplex
- Maintenance of the integrity and consistency of shared data
- High availability of the sysplex

A coupling facility is a special logical partition (LPAR) on a ES/9000 processor, a S/390 9672 or a S/390 9674. For all systems in the sysplex to share data within the coupling facility, they have to be connected to the coupling facility by a new type of channel (coupling facility channels).

Systems in a sysplex that wants to use the coupling facility must have access to a couple data set; *coupling facility resource management* (CFRM) couple data set.

RACF sysplex data sharing and RACF sysplex communication exploit these new MVS/ESA facilities to provide a single image of the RACF database across a sysplex environment by:

- Synchronizing the local RACF data buffers in participating MVS images in a sysplex
- Propagating RACF commands across the the participating members in a sysplex

The new MVS/ESA facilities are also used to enhance the performance of the RACF database in a sysplex environment.

1.2 Optimizing RACF Performance Prior to Data Sharing

To avoid database I/O, RACF buffers RACF data in various ways. Different techniques are used by RACF in different type of operations. The installation has options to specify which technique should be used and how much storage space should be reserved. Figure 1 depicts the techniques that are used by RACF, the type of operation that will use the technique, and the type of storage that is used.

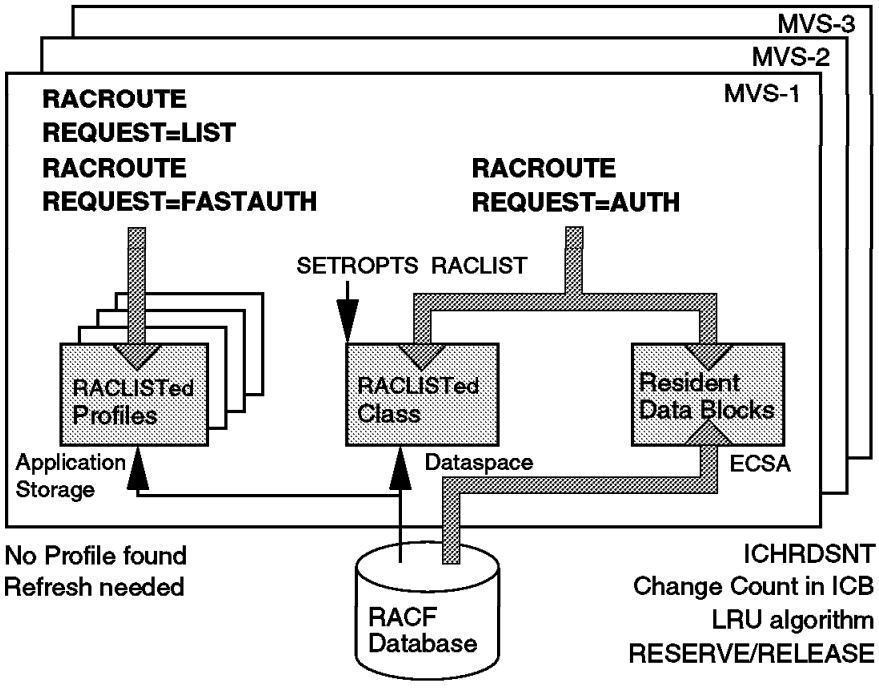

Figure 1. Optimizing RACF Performance Prior to Data Sharing

Each MVS image in a sysplex can specify which technique it will use and how much storage it will reserve.

1.2.1 Resident Data Blocks

In the RACF database name table (ICHRDSNT), an installation can specify the number of resident data blocks for each primary RACF database. For best performance you specify as large a number of buffers as an installation can afford. These resident data blocks keeps any type of RACF data block resident. RACF data blocks could contain:

- RACF profiles
- Block Availability Blocks (BAM)
- Database index blocks

During IPL, RACF obtains the storage for the number of buffers that are specified in the ICHRDSNT from the Extended Common Storage Area (ECSA). In order to maintain a balanced used of the buffer space, the RACF manager keeps track of when each buffer was used last.

The RACF manager does different processing for shared and non-shared databases. When resident data blocks are used for a shared RACF database, like in case of a sysplex environment, RACF has to provide a means of communicating changed RACF data blocks, such as profiles, to the local RACF buffers in each MVS image. This will ensure that all MVS images use the latest level of the RACF data blocks.

When is profile a needed, for example in case of a authorization check, RACF will first read the resident data blocks to find a copy of this profile. If this profile is not found in the resident data blocks, RACF will use the profile in the RACF database and will also copy the data block that contains this profile into the resident data blocks.

1.2.2 RACLISTed Class and RACLISTed Profiles

An installation can optimize performance by carefully deciding whether to use SETROPTS RACLIST or SETROPTS GENLIST for various RACF classes. The RACLIST operand on the SETROPTS command improves performance by copying generic and discrete profiles for the designated general-resource class from the RACF database into a data space. The GENLIST operand results in copying the generic profiles for that general-resource class in a data space.

Prior to RACF 2.1.0, customer applications or MVS subsystems could improve performance by copying selected RACF profiles into the application storage by using the macro RACROUTE REQUEST=LIST. The RACROUTE REQUEST=FASTAUTH could then be used to verify a user′s access to a resource. The RACROUTE REQUEST=FASTAUTH does not gather statistics or issue SVCs. Therefore, use of this RACROUTE macro is recommended only for applications that have stringent performance requirements.

In both circumstances, SETROPTS RACLIST and RACROUTE REQUEST=LIST, if a profile is not found, RACROUTE REQUEST=AUTH and REQUEST=FASTAUTH will return RC=4. The issuing application may treat this as a failure and should provide the appropriate recovery actions. RACF will not find any new profile that is created after the profiles are RACLISTed. Also updates to the profiles are not reflected in the already RACLISTed profiles. The RACLISTed profiles need to be refreshed to reflect the updated profiles, either by:

- A RACF command SETROPT RACLIST (*class_name*) REFRESH
- A RACROUTE REQUEST=LIST ENVIR=DELETE followed by RACROUTE REQUEST=LIST ENVIR=CREATE

Operator commands are usually provided to perform the refresh of application specific RACLISTed profiles. For example, in case of CICS the SECURITY REBUILD transaction performs this sequence of events on each individual CICS system.

RACF commands, like some SETROPTS options that are needed to refresh RACLISTed profiles, are effective only on the systems where they are entered. The operator has to re-enter these commands on every system that is sharing the RACF database and is using RACLISTed profiles.

Prior to RACF 2.1.0, it is the customers responsibility to propagate the information on changed profiles to all the applications and subsystems that are using RACLISTed profiles. If this is not done in a timely fashion, RACF processing can be different on different systems in a sysplex.

1.3 Optimizing RACF Performance With RACF V2.1

Some of the enhancements in the RACF performance optimizing area are related to the exploitation of the enhanced functions in a sysplex environment. Other improvements are available also outside a sysplex implementation.

In RACF sysplex data sharing, the coupling facility is exploited to provide a large buffer for records from the RACF database.

Within the coupling facility, storage is dynamically partitioned into different *structures*. RACF is using *cache* structures as high speed buffers for storing shared data with common read/write access. This high speed buffer is used with the local system buffer, the resident data blocks, to reduce I/O to the RACF database. It also permits RACF to determine more easily if another system in the sysplex has made changes that invalidate records in the local buffer.

Figure 2 depicts the different techniques that can be used by RACF 2.1.0 to optimize the RACF performance. It also shows the type of operation that will use the technique, and the type of storage that is used with this technique.

If an installation uses RACROUTE REQUEST=LIST, they also can improve performance by specifying GLOBAL=YES on the request. GLOBAL=YES stores the RACLIST=LIST results in a data space. This data space can be shared by other applications or subsystems on the same MVS image that issue the same type of request. Additional RACROUTE requests do not access the RACF database; they access the data space built by the first RACROUTE or by the SETROPTS RACLIST command.

The different applications do not need to individually issue RACROUTE REQUEST=LIST deletes followed by RACROUTE REQUEST=LIST creates to refresh the original RACROUTE. The system administrator can do that by issuing a single SETROPTS RACLIST(*class_name*) REFRESH, which references the RACF database to build the RACLIST results and stores them in a new data space. This data space then becomes accessible to any application address space that has issued REQUEST=LIST,GLOBAL=YES for that class. The deletion of the old data space completes this operation.

RACF sysplex communication is used to propagate the RACLIST REFRESH command across the sysplex. For more information on this topic, refer to Chapter 5, “RACLIST Enhancements” on page 39.

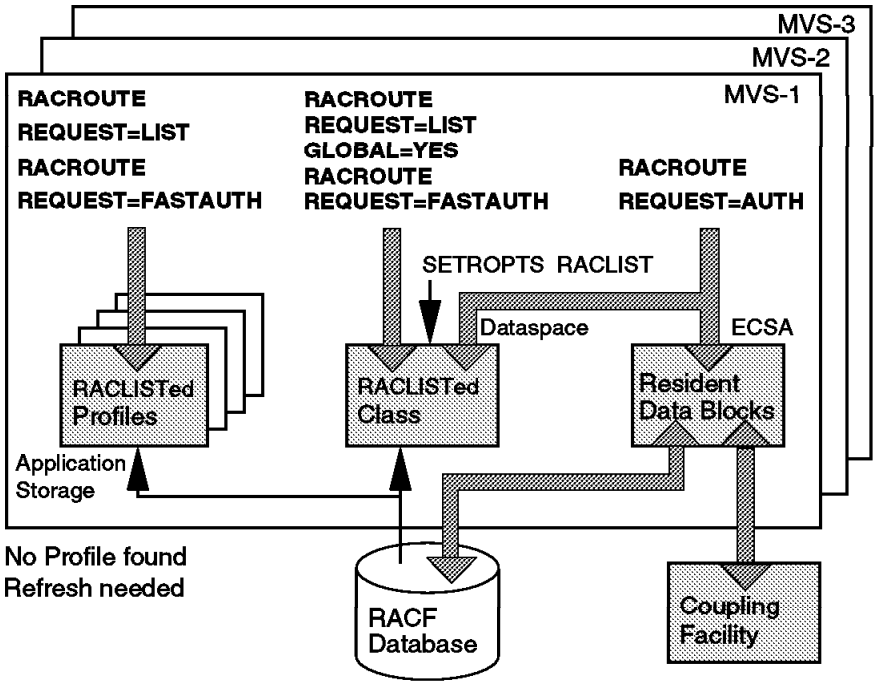

Figure 2. Optimizing RACF Performance in RACF V2R1

The application can still use the RACROUTE REQUEST=LIST facility in a sysplex environment. In this case, the updates in RACF profiles are not reflected in the application storage after a SETROPTS RACLIST(*class_name*) REFRESH command. Every application in every MVS image in the sysplex has to maintain the validity of the profiles by using the delete and create parameter on the RACROUTE REQUEST=LIST facility.

RACF uses a different serialization protocol to replace the RESERVE/RELEASE when RACF is enabled for RACF sysplex data sharing. This protocol uses GLOBAL ENQs to protect the integrity of RACF's data in the RACF database. Refer to 1.5, "Optimizing RACF Performance With Data Sharing Enabled" on page 8 for more detail on this topic.

1.4 RACF Data Buffer Synchronization

Depending on the environment, the RACF manager uses a different technique to synchronize the various RACF local buffers. These environments are:

- Nonshared RACF database
- Shared RACF database
- Data sharing mode utilizing a coupling facility

Nonshared RACF Database: In a *nonshared RACF database*, the RACF manager first searches the resident data blocks for a valid copy of the needed block. If it finds one, it uses it. If it doesn't find a valid copy, the RACF manager obtains an in-storage buffer from the pool of buffers within the resident data blocks, reads the data block from the database into that buffer, and retains the data block in storage after the I/O operation.

When updating a profile, the RACF manager searches the in-storage buffers for a copy of the block that contains this profile. If it doesn't find one, it obtains an

in-storage buffer from the pool of buffers. When a block is updated, RACF always performs an I/O operation to store the new data block on the RACF database to ensure that the database has an up-to-date version of the block.

When getting a buffer from the pool, the RACF manager attempts to get a buffer that is empty. If it finds none, the manager takes the buffer containing the least-recently used block.

Shared RACF Database: The local buffers in the systems that share the RACF database must be synchronized. After a block is updated in the database by one system, the other systems may no longer use this block′s information in their resident data blocks. Synchronization is performed by using *change count arrays* in the *inventory control block* (ICB). The change count in the ICB, corresponding to the block type (profile or index), is updated whenever a block is updated. Index block buffers and profile buffers are marked as *out-of-date* if the change count in the ICB differs from the change count in the in-storage buffer.

Before each resident data block search, the ICB must be read and the change count arrays compared. If a count differs, all blocks for one index level or all data blocks are invalidated.

For performance reasons, the ICB is read whenever RACF needs to do physical I/O to the database, or when RACF can satisfy the request from the resident data blocks but more than two seconds have elapsed since the last time RACF did read the ICB. Therefore, a change made in another system less than two seconds ago might not be picked up instantaneously by other systems.

Data Sharing Mode: In *data sharing mode*, resident data blocks are required both for the primary and backup RACF database. The resident data blocks for the backup RACF database will mainly hold index blocks. The size of the resident data blocks for the backup RACF database is fixed at 20% of the size of the resident data blocks for the primary RACF database.

If the RACF database is split into multiple data sets, each data set (primary or backup) has it own set of resident data blocks.

In the coupling facility, a cache structure must be defined for each set of resident data blocks. This means that a cache structure must exist for each data set in the primary and backup RACF database.

All RACF data, such as profile data, entries from the various index levels, and data in the ICB, is accessed from a copy of the corresponding block in the local buffer (resident data block). If a block cannot be found in the local buffer, it is copied from the cache structure in the coupling facility to the local buffer.

If a block is neither in the local buffer nor in the cache structure of the coupling facility, it is read from the RACF database and stored in the local buffer. Subsequently, the contents of the block is copied into the cache structure in the coupling facility.

If an update to an index or profile data block has to be performed, the block contents are changed in the local buffer while the copy in the cache structure is *invalidated*. Then the changed block is written to the RACF database. When the I/O is successful, the contents of the block are copied into the cache structure of the coupling facility.

To enable synchronization of the local buffers in a sysplex environment (and data sharing enabled), each resident data block used must be associated with a block in the cache structure of the coupling facility. This process of associating a block in the local buffer with a block in the cache structure is called *registering interest*.

If a block in the cache structure becomes invalid, the associated block in each local buffer that has registered interest also becomes invalid. This function is called *cross-invalidation*.

Before accessing a block in the local buffer, RACF uses an XES service to check the block's validity. Figure 3 depicts the synchronization process when RACF data sharing is enabled.

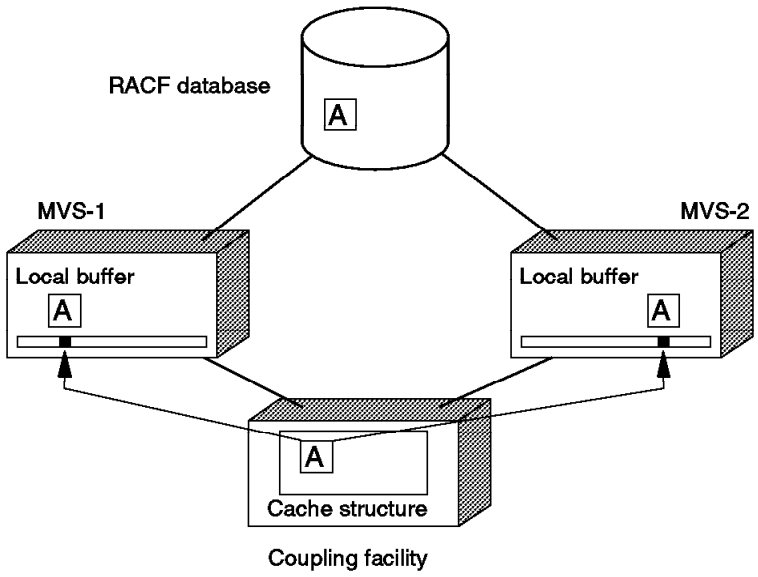

Figure 3. RACF Caching Synchronization

Cache management is dealing with the case all available space in a cache has been occupied and some of this space must be reclaimed to make room for new data to be cached. The RACF manager as well as the coupling facility cache manager use a *least recently used* (LRU) algorithm. The local buffers (resident data blocks) are managed by the RACF manager. The cache structures are managed by the coupling cache manager.

The LRU algorithm assumes that the probability that data will be needed again is decreasing with the time the data has not been referenced. Therefore, the blocks in the cache that have not been referenced for the longest time are discarded to make room for new data to be cached.

When a block in the local buffer is reclaimed by the LRU algorithm, the association between this block and the corresponding block in the cache structure of the coupling facility must be dropped. Interest is de-registered.

When a block in the cache structure of the coupling facility is reclaimed by the cache manager, the associated blocks in the local buffers are cross-invalidated.

Chapter 3, “Sysplex Communication and Data Sharing” on page 21 describes how to implement RACF sysplex communication and RACF sysplex data sharing.

1.5 Optimizing RACF Performance With Data Sharing Enabled

If all systems is a sysplex must share the same RACF database, some performance problems could arise if RACF data sharing is not enabled. As the number of systems increases, the increased hardware RESERVE on the RACF database could adversely influence the performance of the I/O to the RACF data sets. If the RACF database is updated frequently, the performance advantantage of a local buffer (resident data blocks) will decrease considerably.

RACF sysplex data sharing enables RACF to eliminate these problems.

Rather than using RESERVE/RELEASE, RACF will use *general resource serialization* (GRS) ENQ for the serialization of accesses to data in the RACF database. RESERVE is always excluding all other systems that share the DASD volume from any access; ENQs allow any number of READ operations to take place concurrently. GRS uses *cross system coupling facility* (XCF) for the communication between the systems.

In data sharing mode, the coupling facility is exploited to provide a large buffer for records from the RACF database, allowing a decrease in the I/O rate to the RACF database.

Data sharing mode also permits RACF to determine more easily if another system has made changes that invalidate records in the local buffer. In data sharing mode, only the block that has been updated needs to be invalidated instead of all blocks in a index level or all data blocks. Depending on the number of systems sharing the RACF database and the amount of update activity that is taking place, this new synchronization technique has potential for a considerable improvement in the performance of the local buffers (resident data blocks).

Chapter 2. RACF Version 2 Release 1 Installation

This chapter describes an installation scenario for RACF 2.1.0. This scenario is based on the installation experience in a *parallel sysplex* environment.

2.1 General Product Information

RACF 2.1.0 is a new version which replaces all previous versions. The product can be ordered through the following ordering procedures:

- As a stand-alone product
- Within an MVS Custom-Built Installation Process Offering (CBIPO)
- Within an MVS Custom-Built Product Delivery Offering (CBPDO)
- Through the following CustomPac offerings :
  - ProductPac
  - ProductPac/E
  - FunctionPac
  - SystemPac

  Note: Not all CustomPac offerings are available in all countries.

Whichever orderable you choose, each supplies its own installation documentation and procedures and may refer to the *RACF Program Directory* of the stand-alone product. This document is supplied with all orderables.

Table 7. Component ID

Component ID	Program Number	Component name	V/R/M	FMID
5752XXH00	5695-039	RACF	2.1.0	HRF2210

2.2 Product Considerations

When planning the installation, please be particularly aware of the following:

1. RACF 2.1.0 cannot be installed or operated on MVS/370 or MVS/XA systems; the product requires MVS/ESA Version 4 Release 2 or higher.
2. RACF 2.1.0 operates only with the *restructured RACF database*, which was introduced with RACF Version 1 Release 9.

   If you still have a non-restructured RACF database, you will need the *RACF Program Directory* for RACF 1.9 for information how to migrate to a restructured RACF database.
3. Assemblies of programs which use the RACROUTE macro must be performed with Assembler H or equivalent; Assembler F will *not* work anymore.
4. If you have not taken precautions, your *RACF started procedures table* (module ICHRIN03 in LPALIB) will be replaced by a dummy table with no entries.

Although RACF 2.1.0 now supports dynamic changes to the started procedures table, RACF still requires the presence of at least a dummy module ICHRIN03 at IPL time.

It you start using the *dynamic started procedures table* support (by activating the resource class STARTED along with your desired profiles), it is recommended to maintain critical entries, such as for JES, VTAM, TSO, and so on, in module ICHRIN03 in parallel. Please see *RACF System Programmer's Guide* for more information.

The RACF Data Security Monitor (DSMON) can be used to create a *started procedures table report*. When the STARTED class is active, information is retrieved from the RACF database. If not, the report is created from the active ICHRIN03 module. See Appendix A, "Started Procedure Tables Report" on page 81 for the sample JCL how to call DSMON and for a sample output report.

The DSMON started procedures table report can be used to help recreate the ICHRIN03 module (in case you lost the source code) or to create definitions for the resource class STARTED. This will be covered later in this redbook.

2.3 Driving System Requirements

The system with which you install RACF 2.1.0 (the so-called *driving system*) requires the following products (or their equivalent) for a successful installation of the product :

Table 8. Driving System Requirements

Program Product	**Program Number**	**Minimum Version/Release/Service Level**
SMP/E	5668-949	V1 R7 on Service Level 17.20, that is : UR40251 for FMID HMP1700 UR40252 for FMID JMP1701 UR40255 for FMID JMP1711
Linkage Editor or Program Binder	---	Supplied with one of the following : MVS/DFP DFSMSdfp
EITHER		
Assembler H	5668-962	V2 R1 with PTFs UL90149, UL33954 and UL67573
OR		
High Level Assembler	5696-234	V1 R1

2.4 ITSO Poughkeepsie Sysplex Environment

Figure 4 depicts the sysplex configuration in which the installation procedures are tested. The configuration consists of six MVS images and two coupling facilities. Three MVS images are running in a 9021-982, two MVS images are running in a 9672-E03, and one is running in a 9672-E01. All DASD in the sysplex is accessible by all the MVS images.

The two coupling facilities are running in the 9672-E03.

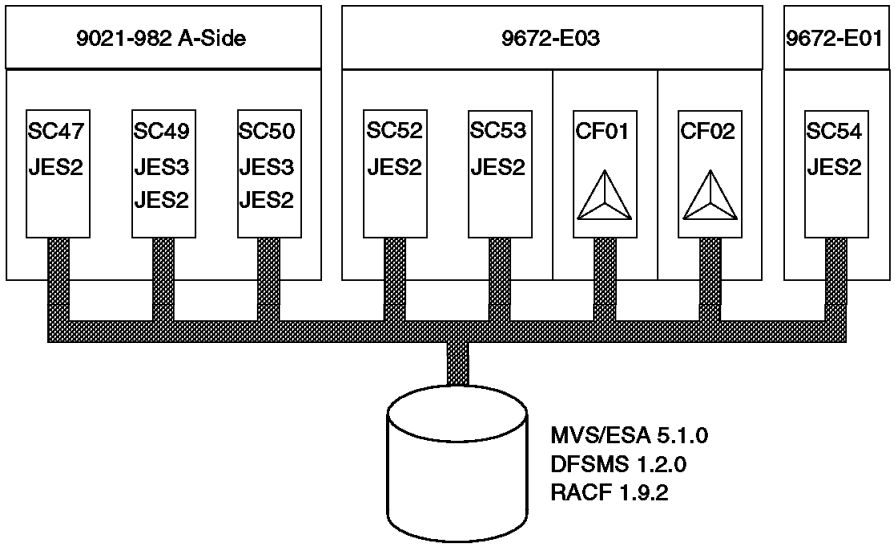

Figure 4. ITSO Poughkeepsie Sysplex Environment

The following abbreviations are used in Figure 4:

SCnn Stands for an MVS system running in a partition on the respective processor.

CFnn Stands for a coupling facility, which runs the *coupling facility control code* (CFCC).

JESn Stands for the Job Entry Subsystem (JES) type running on the respective MVS. The systems SC49 and SC50 contain a JES3 as the primary and a JES2 as a secondary Job Entry Subsystem. All JES2s in the sysplex operate in a *multi-access spool* (MAS) environment.

2.5 ITSO Poughkeepsie SMP/E Structure

Figure 5 depicts the SMP/E structure that was used to install RACF 2.1.0 in the sysplex environment that was described in 2.4, “ITSO Poughkeepsie Sysplex Environment” on page 11.

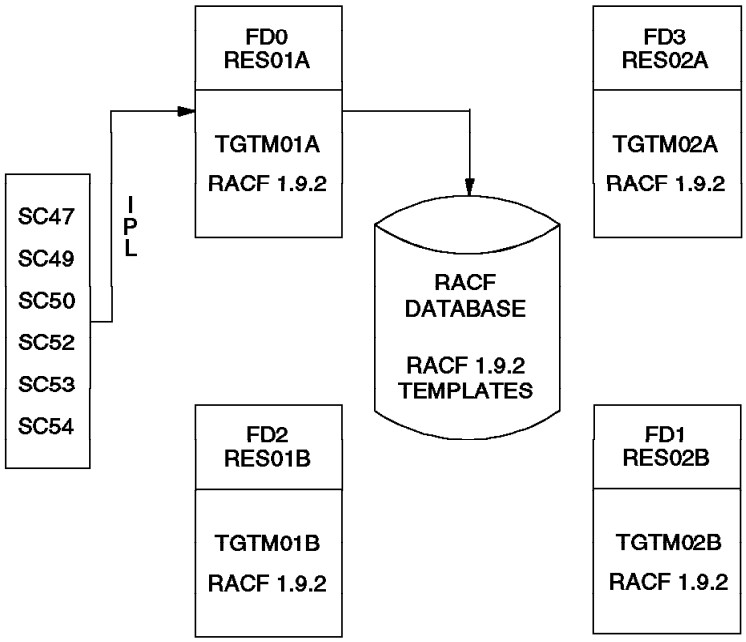

Figure 5. SMP/E Structure

The following abbreviations are used in Figure 5:

RESnnn Stands for a SYSRES volume, which contains :

- The TARGET libraries
- The corresponding TARGET CSI data set
- Other SMP/E data sets (SMPMTS, SMPSTS, SMPSCDS, SMPLTS, SMPLOG and SMPLOGA) related to that TARGET CSI

TGTMnnn Stands for the TARGET ZONE NAME.

Each DDDEF in the TARGET ZONE uses the VOLUME and UNIT parameter to point to the respective data set on that volume.

SMP/E maintenance is performed by installing the SYSMODs into just ONE TARGET CSI on a SYSRES volume which is *not* IPLed at that time. When maintenance installation has completed, that SYSRES volume will become the IPL volume for the next IPL.

When everything works fine, the updated volume is copied to its respective backup volume. The TARGET CSI on the backup volume is updated and all VOLUME sub entries in the DDDEFs are modified to point to the backup volume′s data sets.

The rest of the SMP/E environment consists of one GLOBAL zone, one DISTRIBUTION zone and the corresponding data sets, like SMPPTS, the SMPLOG and SMPLOGA for the GLOBAL CSI and so on.

Each RESnnB volume is a backup copy of the respective RESnnA volume. All systems are IPLed from the volume RES01A. All four SYSRES volumes currently contain MVS/ESA Version 5.1 along with DFSMS 1.2.

The data sets SYS1.PARMLIB, SYS1.IPLPARM as well as the master catalog are shared between all systems in the sysplex and reside on different DASD volumes. All systems use the &SYSNAME. naming technique for the LOGREC, STGINDEX and PAGE data sets, which again reside on different DASD volumes.

All TARGET data sets are cataloged in the (shared) master catalog using the *indirect cataloging* technique (UNIT=0000 and VOLUME=******) to be able to have multiple data sets with the same name (for example: all four RESnnn volumes contain a SYS1.LINKLIB).

2.6 Product Installation

This topic describes the steps to install the stand-alone product along with the available PTF maintenance into the TARGET ZONE **TGTM02A** on the SYSRES volume **RES02A**.

1 Obtain RACF 2.1.0 and the *RACF Program Directory*.

2 Obtain the latest *preventive service planning* information, the so called PSP bucket, from your IBM representative.

Table 9. Preventive Service Planning information for RACF 2.1.0

Upgrade	Subset
RACF210	HRF2210

Check the PSP data for any information pertinent to the installation of RACF 2.1.0.

3 Check data set space information.

If you already have a RACF installed on your system, no new libraries are needed to install RACF 2.1.0; existing libraries will be updated.

Please refer to the *RACF Program Directory* for more information about data set space information.

4 SMP/E Product RECEIVE

Perform an SMP/E RECEIVE for the function HRF2210. The following SMP/E control statements were used:

```
SET BDY(GLOBAL) .
  RECEIVE S(HRF2210) .
```

5 SMP/E Service RECEIVE

Perform an SMP/E RECEIVE for all available PTFs (from the IBM supplied cumulative PTF tape or from other sources) along with the HOLDDATA. The following SMP/E control statements were used:

```
SET BDY(GLOBAL) .
  RECEIVE SYSMODS HOLDDATA SOURCEID(ssssssss) .
```

The SOURCEID "ssssssss" is a (maximum) 8-character name which SMP/E assigns to all the SYSMODs being processed. This allows you to identify multiple SYSMODs through just one name.

6 SMP/E APPLY CHECK

Perform an SMP/E APPLY CHECK run of both the function HRF2210 and all applicable PTFs for HRF2210. The following SMP/E control statements were used:

```
SET BDY(TGTM02A) .
  APPLY
  CHECK
  SELECT(HRF2210)
  FORFMID(HRF2210)
  FUNCTIONS PTFS
  GROUPEXTEND(NOAPARS,NOUSERMODS)
  BYPASS(HOLDSYSTEM,HOLDUSER,HOLDCLASS(UCLREL,ERRREL)) .
```

Warning: Carefully investigate the output of the APPLY CHECK run:

- Check the SMP/E File Allocation Report to verify that the correct data sets on the correct DASD volumes were used.
- Check the SMP/E Sysmod Status Report for any USERMOD which will be deleted during the real APPLY run.
- The APPLY CHECK job should end with a return code not higher than 4.

7 ICHRIN03 considerations

Warning: The SMP/E APPLY run will replace your existing copy of module ICHRIN03 on your target system with a dummy module supplied with RACF 2.1.0 (if you have not taken precautions to avoid that).

Now it is a good time to SAVE that copy of ICHRIN03 to a library which is not going to be touched by SMP/E during the APPLY run.

8 SMP/E APPLY

Perform an SMP/E APPLY run of both the function HRF2210 plus all the PTFs for HRF2210. The same SMP/E control statements as for the APPLY CHECK run may be used; the keyword **CHECK**, however, must be deleted.

The APPLY job should end with a return code not higher than 4.

Note

At this point in time, the SMP/E part of installing the product RACF 2.1.0 along with its applicable maintenance on volume RES02A has completed. The product can now be customized.

9 IRRMIN00

To upgrade the existing RACF 1.9.2 database to the RACF 2.1.0 level, program IRRMIN00 on the RACF 2.1.0 level must be used. The program resides in SYS1.LINKLIB and must be APF-authorized on the driving system. The following JCL can be used :

```
//UPDRACFP EXEC PGM=IRRMIN00,PARM=UPDATE
//         SET VOL=RES02A
//         SET UNT=SYSALLDA
//STEPLIB  DD DISP=SHR,DSN=SYS1.LINKLIB,
//            UNIT=&UNT,VOL=SER=&VOL
//SYSPRINT DD SYSOUT=*
//SYSTEMP  DD DISP=SHR,DSN=SYS1.MODGEN(IRRTEMP1),
//            UNIT=&UNT,VOL=SER=&VOL
//SYSRACF  DD DISP=SHR,DSN=SYS1.RACFESA
```

Notes:

a. PARM=UPDATE must be used to update an existing RACF database with the new RACF 2.1.0 templates. All existing profiles will remain intact.

   Do NOT use PARM=NEW on an existing RACF database; this will delete all data!

b. The DD statement “SYSRACF” points to the *existing* primary RACF data set.

c. The DD statement “SYSTEMP” points to the member containing the new RACF 2.1.0 templates.

Repeat this step for your BACKUP RACF data set.

10 Database Range Table

As the ITSO Poughkeepsie does not use multiple RACF databases, it was not necessary to create a RACF Database Range Table. Appendix C, “Database Range Table - Sample JCL” on page 87 shows sample JCL to establish exactly the RACF-supplied default Database Range Table.

For more information about the RACF Database Range Table, please refer to the *RACF System Programmer's Guide*.

11 Database Name Table

The data set names of the RACF data sets must be identified to RACF itself in one of the two following manners :

- Through a valid Master JCL member (MSTJCLxx)
- Through module ICHRDSNT

As it was the intention to activate both RACF sysplex communication as well as RACF sysplex data sharing (at a later stage), the only way to specify these options is through the database name table (ICHRDSNT). The load module ICHRDSNT must reside in SYS1.LINKLIB or any other library concatenated to SYS1.LINKLIB through the LNKLSTxx member in SYS1.PARMLIB.

Refer to *RACF System Programmer's Guide* for an *emergency database names table* procedure that allows you to specify alternate database names if you cannot access your normal RACF database.

The ITSO Poughkeepsie uses an SMP/E USERMOD to specify the appropriate RACF database data set names in SYS1.LINKLIB(ICHRDSNT). See Appendix D, “Database Name Table - Sample JCL” on page 89 for the sample JCL.

12 PARMLIB Fix List

To improve CICS performance, the active IEAFIXxx member in SYS1.PARMLIB was updated with the following modules, residing in data set SYS1.LPALIB :

- ICHRFC00
- IGC0013{
- ICHRFR00
- ICHSFR00

13 PARMLIB VLF

To improve RACF performance, modifications were made to the active COFVLFxx member in SYS1.PARMLIB.

- To activate Group Tree in Storage (GTS) support, the following class was added :

```
CLASS NAME(IRRGTS)
      EMAJ(GTS)
```

- To activate the support to save ACEEs in VLF, the following class was added :

```
CLASS NAME(IRRACEE)
      EMAJ(ACEE)
```

- To activate OpenEdition MVS performance improvements, the following classes were added :

```
CLASS NAME(IRRGMAP)
      EMAJ(GMAP)

CLASS NAME(IRRUMAP)
      EMAJ(UMAP)
```

14 Started Procedures Table

Copy the saved version of the started procedures table (ICHRIN03) back into the target system's SYS1.LPALIB data set.

If you forgot to save module ICHRIN03 before the APPLY run, or if the source code is not available anymore, please see Appendix E, "Started Procedures Table - Sample JCL" on page 91 for sample JCL.

15 Before IPL

We have now reached the following situation:

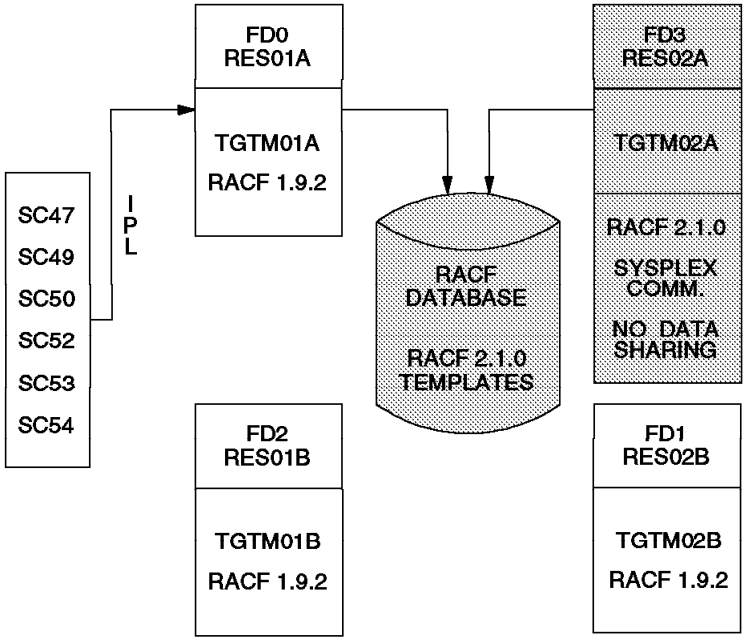

Figure 6. SMP/E Structure - RACF 2.1.0 Installed

- The product RACF 2.1.0 along with its available PTF maintenance is installed (APPLIED) on volume RES02A.
- The existing RACF database has been updated with the new RACF 2.1.0 templates.
- On volume RES02A, the RACF database name table (ICHRDSNT) points to the existing and migrated RACF database; systems IPLing from this volume will enable RACF to use sysplex communication.
- The *RACF class descriptor table* (ICHRRCDX) and the *RACF router table* (ICHRFR0X) did not require any modification. Please see *RACF Macros and Interfaces* for more information about the IBM-supplied Class Descriptor Table and about the characteristics of each class.
- *Dynamic parsing* was already setup to start automatically at IPL time.
- The RACF panels, CLISTs, skeletons and messages were already concatenated to the respective DD statements in the LOGON procedure.

16 Activating the RACF subsystem

The RACF subsystem provides an additional execution environment for the RVARY command and provides support for APPC persistent verification. Starting the subsystem is optional but recommended.

Although it was already available in a previous version of RACF, the RACF subsystem is becoming more important now, especially in a sysplex environment. When the RACF subsystem is active, RVARY can be issued as an MVS operator command with the appropriate prefix, as specified by the installation. You can continue to enter RVARY as a regular TSO command.

The RACF subsystem address space is identified as a standard MVS subsystem. The RACF subsystem reads startup parameters from the IEFSSNxx member of SYS1.PARMLIB and the PARM keyword on the EXEC statement in the subsystem procedure. An installation can choose to use the MVS subsystem convention of assigning a unique subsystem prefix or they can use the unique subsystem name, followed by a blank, as the prefix for the RACF subsystem.

To activate the RACF subsystem, you must:

- Update the IEFSSNxx member of SYS1.PARMLIB. This member must be updated to indicate that the RACF subsystem is a valid subsystem in the installation. This member also identifies the subsystem's command prefix used in issuing RACF operator commands, and an optional command prefix scope.

  An installation can choose to have RACF register the command prefix with the MVS *command prefix facility* (CPF). CPF allows an operator or authorized application to enter a RACF command from any system in the sysplex and route that command to run on another system in the sysplex. The command responses come back to the originating system console.

  For more information on CPF, see:

  - *RACF System Programmer's Guide*
  - *MVS/ESA Application Development Guide: Authorized Assembler Language Programs* for MVS Version 4
  - *MVS/ESA Programming: Authorized Assembler Services Guide* for MVS Version 5

  Refer to *RACF Command Language Reference* for information on how to use the subsystem command prefix.

  Although it is recommended, we did not specify a *scope* for the command prefix at this stage of our sysplex implementation. We used the following specification in the IEFSSNxx member of SYS1.PARMLIB:

  ```
  RACF,IRRSSI00,'#'
  ```

  RACF is the subsystem name and # is the command prefix. Because no scope is specified, the command prefix is not registered with CPF.

- Update the SCHEDxx member of SYS1.PARMLIB. We added the entry as suggested in *RACF System Programmer's Guide*.

- Assign a RACF user ID to the RACF subsystem. The RACF subsystem cannot be initialized if a valid RACF user ID is not assigned to it. The PROC name for the RACF subsystem must be the same as the name

used in IEFSSNxx. We added an entry for the RACF subsystem into the *started procedures table* (ICHRIN03).

After IPL and initialization of the new RACF environment, we added a profile for RACF in the STARTED class through the following command:

```
RDEFINE STARTED RACF.* STDATA(USER(STC) GROUP(SYS1) PRIVILEGED(NO)
   TRUSTED(NO) TRACE(NO)
```

Note: It is not necessary to mark the entry *privileged* or *trusted*.

- Review the RACF PROC provided in SYS1.PROCLIB. JCL to activate the RACF subsystem is provided in SYS.PROCLIB(RACF). We entered the following statements:

```
//RACF PROC
//RACF EXEC PGM=IRRSSM00,REGION=0M
```

17 IPL

All systems in the sysplex can now be IPLed from volume RES02A. During the IPL phase, the following RACF-related messages can be observed:

```
ICH525I INSTALLATION ROUTER TABLE PROCESSED
ICH508I ACTIVE RACF EXITS: NONE
ICH524I INSTALLATION CLASS DESCRIPTOR TABLE PROCESSED
ICH559I MEMBER SCnn ENABLED FOR SYSPLEX COMMUNICATIONS
IRRX005I MEMBER SCnn IS IN NON-DATA SHARING MODE.
ICH520I RACF 2.1.0  IS ACTIVE.
ICH531I RACF DATA SET ALLOCATION/DEALLOCATION INTERFACE IS ACTIVE.

START RACF,SUB=MSTR

IEF196I IEF695I START RACF     WITH JOBNAME RACF     IS ASSIGNED TO
IEF196I USER STC     , GROUP SYS1

IEF695I START RACF     WITH JOBNAME RACF     IS ASSIGNED TO USER STC
 , GROUP SYS1

IEF403I RACF - STARTED - TIME=14.17.38

#IRRB001I RACF SUBSYSTEM 2.1.0 IS ACTIVE.
```

The character "#" in the IRRB001I message is the command prefix character which has been assigned to the RACF subsystem in the active IEFSSNxx member in SYS1.PARMLIB. When the RACF address space is started, the user ID STC and the group SYS1 are assigned to it through ICHRIN03.

18 Define and activate the STARTED class

When RACF 2.1.0 is up and running, resource definitions for the STARTED class can be created and the class can then be activated. See Appendix B, "Convert ICHRIN03 to STARTED Class Profiles" on page 83 for a description of how to accomplish that.

19 SMP/E ACCEPT CHECK

Perform an SMP/E ACCEPT CHECK run of both the function HRF2210 and all APPLIED PTFs for HRF2210. The following SMP/E control statements can be used:

```
SET BDY(MVSDLIB) .
  ACCEPT
  CHECK
  SELECT(HRF2210)
  FORFMID(HRF2210)
  FUNCTIONS PTFS
  GROUPEXTEND(NOAPARS,NOUSERMODS)
  BYPASS(HOLDSYSTEM,HOLDUSER,HOLDCLASS(UCLREL,ERRREL)) .
```

Warning: Carefully investigate the output of the ACCEPT CHECK run:

- Check the SMP/E File Allocation Report to verify that the correct data sets on the correct DASD volumes will be used during the real ACCEPT step.
- The ACCEPT CHECK job should end with a return code not higher than 4.

20 SMP/E ACCEPT

Perform an SMP/E ACCEPT run. The same SMP/E control statements as for the ACCEPT CHECK run may be used; the keyword **CHECK**, however, must be deleted.

The ACCEPT job should end with a return code not higher than 4.

Chapter 3. Sysplex Communication and Data Sharing

This chapter describes how to enable the RACF sysplex communication and RACF sysplex data sharing facilities. It also provides a more detailed description of these new facilities.

Chapter 1, "Introduction to RACF Version 2 Release 1" on page 1 provided a brief overview of RACF sysplex data sharing and RACF sysplex communication facilities and their benefits for customer installations. Chapter 5, "RACLIST Enhancements" on page 39 provides a detailed description of the RACLIST enhancements and how they work together with RACF sysplex communication. That chapter also describes some sample programs on how to demonstrate the new facilities.

3.1 Sysplex Considerations

The goal of RACF in a sysplex is to provide security for all resources of all systems in a comprehensive and centralized way. RACF allows the installation to use a coupling facility and shared RACF data to help manage the security of resources for all systems in a sysplex.

A coupling facility allows MVS and other software to share data concurrently among multiple systems in the sysplex with the goal of maintaining a single system image. A sysplex with a coupling facility significantly changes the way systems can share data. The technology that makes high performance sysplex data sharing possible is a combination of hardware and software services. *Data sharing* is the ability of concurrent subsystems or application programs to directly access and change the same data while maintaining system integrity.

RACF is designed so that its database can be shared between processor complexes while data integrity is maintained. Because of the need to serialize RACF database processing, there may be I/O contention. *RACF sysplex data sharing* is designed to address the problems that can occur when many systems share a RACF database.

Many SETROPTS options (for example SETROPTS CLASSACT) automatically take effect across sharing systems when the inventory control block (ICB) of the master primary RACF database is read on the other systems. Other SETROPTS commands (for example SETROPTS RACLIST) must be issued explicitly on all sharing systems. *RACF sysplex communication* introduces command propagation for certain commands (for example SETROPTS RACLIST). You no longer have to issue these commands on each system sharing the database. Instead, you can issue a command once, and RACF propagates it to the other systems in the sysplex. For more information on command propagation, see *RACF System Programmer's Guide* and *RACF Security Administrator's Guide*.

If all the requirements that are listed in 3.2.1, "RACF Sysplex Communication" on page 22 are met, the sysplex can be enabled for RACF sysplex communication. When the system is enabled for RACF sysplex communication, it uses cross-system coupling facilities (XCF) to join the RACF data sharing group, IRRXCF00. The data sharing group facilitates communication between systems in the sysplex enabled for sysplex communication. There is only one data sharing group per sysplex.

When RACF is enabled for sysplex communication, it allocates in-storage buffers for the backup database (20% of the number allocated for the primary database), to reduce I/O to the backup device. Additionally, a minimum of 50 buffers are used for the primary database. As a consequence, the installation may notice an increased Extended Common Storage Area (ECSA) usage for the in-storage buffers.

Once RACF sysplex communication has been activated, RACF can be in one of the following modes:

3.1.1.1 Non-Data Sharing Mode

RACF does not make use of the coupling facility and uses the normal RESERVE/RELEASE serialization protocol for accessing the RACF database.

The RACF database can be shared with systems having RACF 1.9 or 1.9.2 installed (including VM systems).

3.1.1.2 Data Sharing Mode

In this mode, RACF starts exploiting the coupling facility. A large buffer in the coupling facility is allocated to contain records from the RACF database, I/O operations to the physical database on DASD will decrease. A new serialization protocol uses GLOBAL ENQs instead of RESERVE/RELEASE for accessing the RACF database. The RACFDS (**D**ata **S**haring) address space is started automatically and stays alive until the next IPL, even if RACF is placed back into non-RACF sysplex data sharing mode.

Warning: Once in RACF sysplex data sharing mode, the database can *not* be shared with *any* other system that does not participate in this data sharing group. This also excludes systems from other data sharing groups, even if they are in data sharing mode.

3.1.1.3 Read-Only Mode

A system enters this mode when RACF sysplex data sharing mode has been requested and the coupling facility cannot be used because of some reason. Other systems may remain to operate in RACF sysplex data sharing mode. Database updates are *not* allowed on a system in read-only mode. When the problem with the coupling facility is solved, RACF will try to re-enter RACF sysplex data sharing mode.

3.2 Requirements

The following sections describe the requirements that must be met before the sysplex facilities can be enabled.

3.2.1 RACF Sysplex Communication

To be able to use RACF sysplex communication (without RACF sysplex data sharing) on a system in an existing sysplex, the following requirements must be fulfilled on that system:

- MVS Release 4.2 or higher must be installed.
- RACF 2.1 must be installed.
- All systems must *not* be in XCF-local mode.

- The major name SYSZRACF must *not* be in any GRS Exclusion Resource Name List (RNL).
- The RACF database must reside on a shared DASD.
- The RACF Database Name Table (ICHRDSNT) and the Class Descriptor table (ICHRRCDE) must be compatible on all systems.
- All systems must use the same Database Range Table (ICHRRNG).

3.2.2 RACF Sysplex Data Sharing

All systems within an existing sysplex that must participate in RACF sysplex data sharing must met the following requirements (besides the ones for RACF sysplex communication):

- MVS/ESA Version 5.1 (or higher) must be installed.
- RACF 2.1 must be installed.
- RACF sysplex communication must be enabled.
- Valid “structures” must be defined in a *coupling facility resource manager* (CFRM) policy.
- The CFRM policy containing these RACF structures must be *active*.
- All participating systems must be able to connect to all required structures.

Systems within the sysplex that do not participate in RACF sysplex data sharing must have their own RACF database.

3.3 Enabling RACF Sysplex Communication

RACF sysplex communication can only be enabled by modifying the flag byte of the first entry in the RACF database name table module ICHRDSNT. Please see Appendix D, “Database Name Table - Sample JCL” on page 89 for how to accomplish that.

Activating this modification requires an IPL of the systems.

3.4 Enabling RACF Sysplex Data Sharing

Enabling RACF sysplex data sharing consists of multiple steps:

1 Define the coupling facility structures

Before RACF sysplex data sharing can be started, a so-called “structure” must be defined in the *coupling facility resource manager* (CFRM) policy for each primary and backup RACF database name.

Such a structure is a high-speed cache buffer in the coupling facility. RACF uses non-persistent cache structures, which means that buffers get deallocated when all systems have disconnected from them. The buffer acts as any normal cache storage: it contains data which any system has retrieved from the RACF database and as such, avoids further access to the database when any system has the need to retrieve the same data.

The initial estimate for the primary *structure size* can be calculated with the following formula:

```
(a x 4) + (b x 4 x n)

Where :  a  = The number of data buffers specified
             for the primary database
         4  = The size of each buffer in kilobytes
         b  = Ten percent of a
              (to allow different data reference patterns)
         n  = The number of members in the sysplex
```

The coupling facility storage required for a backup RACF database is 20% of the value for the primary database.

The ideal case is that the primary structure is large enough to contain the entire primary RACF database.

For the ITSO 6-way sysplex with 255 data buffers specified for the primary RACF database, the initial structure sizes were estimated as follows:

Primary (255 x 4) + (26 x 4 x 6) = 1644KB

Backup (0.2 x 1644) = 329KB

Define the structure names in the following format: IRRXCF00_tnnn

t P for primary or B for backup database

nnn The RACF database sequence number

The sequence number is retrieved from the *RACF data set descriptor table* (ICHPDSDT). This table descibes the primary and backup RACF data sets and is created at IPL time with information from the customer-provided load module *database name table* (ICHRDSNT). The sequence number is assigned by RACF in the data set descriptor table according to the sequence in which the RACF data sets are defined in the data set name table. Both sequence numbers, for the primary and the backup, should start with 001.

The structures must be defined with the *administrative data utility* IXCMIAPU, which is described in *MVS/ESA Setting up a Sysplex*. Refer also to Appendix F, “Administrative Data Utility - Sample JCL” on page 93 for how to code IXCMIAPU, or have a look at data set “SYS1.SAMPLIB,” member IXCCFRMP.

2 Activate the coupling facility structures

Once the structures have been defined, the CFRM Policy which contains them must be activated by issuing the following MVS command:

```
SETXCF START,POLICY,TYPE=CFRM,POLNAME=CFRM01
```

Now you can display the policy with the following MVS command:

```
D XCF,POLICY,TYPE=CFRM
```

You can also display all structures:

```
D XCF,STRUCTURE,STRNAME=ALL
```

Or display a specific structure:

```
D XCF,STRUCTURE,STRNAME=IRRXCF00_P001
```

3 Activating RACF sysplex data sharing mode

Once the policy and the structures are all active, RACF can be placed in RACF sysplex data sharing mode by issuing the following RACF command:

```
#RVARY DATASHARE
```

Note: The character # in the RVARY DATASHARE command is the variable subsystem prefix character. This prefix character is discussed in 2.6, “Product Installation” step 16 “Activating the RACF subsystem” on page 18.

This will display the following message:

```
*nnn ICH703A ENTER PASSWORD TO SWITCH RACF MODE JOB=RACF     USER=STC
```

This message informs you to confirm the #RVARY command with the password which has been defined for the RACF *mode switch*. If you have never defined such a password (by issuing a SETROPTS RVARYPW command), the SWITCH password is set to the word *YES*.

Having done that, RACF responds with the following messages:

```
IEE600I REPLY TO nnn IS;SUPPRESSED

ICH15019I INITIATING PROPAGATION OF RVARY COMMAND TO MEMBERS OF RACF
DATA SHARING GROUP IRRXCF00.
```

This informs you that the RVARY command is propagated to all other members of the sysplex.

Some time later, the system will respond with message:

```
IRRX000I MEMBER xxxx IS IN DATA SHARING MODE.
```

This is the final RACF message telling you that the member of the sysplex on which you issued the command (xxxx) is now in data sharing mode. Due to command propagation, all other systems should have switched as well.

You can verify that by issuing the command:

```
ROUTE *ALL,#RVARY LIST
```

All members of the sysplex should answer with a message similar to the following:

```
ICH15013I RACF DATABASE STATUS:
 ACTIVE   USE    NUMBER   VOLUME     DATASET
 ------   ---    ------   ------     -------
  YES     PRIM      1     TOTSM1     SYS1.RACFESA
  YES     BACK      1     TOTRS1     SYS1.RACF.BKUP1
 MEMBER xxxx     IS IN DATA SHARING MODE.
ICH15020I RVARY COMMAND HAS FINISHED PROCESSING.
```

If you now issue the command

```
D XCF,STRUCTURE,STRNAME=IRRXCF00_P001
```

again, you will see that the defined RACF structures now show as allocated.

```
IXC360I  17.17.54  DISPLAY XCF
STRNAME: IRRXCF00_P001
 STATUS: ALLOCATED
  POLICY SIZE    : 1644 K
  REBUILD PERCENT: N/A
  PREFERENCE LIST: CF01     CF02
  EXCLUSION LIST IS EMPTY

 ACTIVE STRUCTURE
 ----------------
  ALLOCATION TIME: 08/04/94 17:03:28
  CFNAME         : CF01
  COUPLING FACILITY: 009672.IBM.02.000000040104
                     PARTITION: 1   CPCID: 00
  ACTUAL SIZE    : 1792 K
  STORAGE INCREMENT SIZE: 256 K
  VERSION        : A9ADCD40 9F715080
  DISPOSITION    : DELETE
  ACCESS TIME    : 0
  MAX CONNECTIONS: 32
  # CONNECTIONS  : 6

  CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
  ---------------- -- -------- -------- -------- ---- --------
  IRRP001@SC47     01 00010002 SC47     RACFDS   002C ACTIVE
  IRRP001@SC49     02 00020002 SC49     RACFDS   001A ACTIVE
  IRRP001@SC50     04 00040002 SC50     RACFDS   0036 ACTIVE
  IRRP001@SC52     06 00060002 SC52     RACFDS   002C ACTIVE
  IRRP001@SC53     03 00030002 SC53     RACFDS   0017 ACTIVE
  IRRP001@SC54     05 00050002 SC54     RACFDS   0019 ACTIVE
```

Please note that the *actual size* of the allocated structures may differ from the specified size due to an upward rounding to a multiple of 256KB.

4 Default to RACF sysplex data sharing mode

To enable RACF to initialize automatically in RACF sysplex data sharing mode at IPL time, the flag byte of the first entry in the RACF data set name table (ICHRDSNT) may be modified. Please see Appendix D, “Database Name Table - Sample JCL” on page 89 for how to accomplish that.

3.5 Coupling Facility Recovery

After enabling the RACF sysplex data sharing facility the following error recovery scenarios for the coupling facility are tested:

1. Rebuild a RACF structure by using a MVS REBUILD command.
2. Removal of one coupling facility through a hardware CONFIG command.
3. Loss of power in one coupling facility.
4. Loss of connectivity between one system and a coupling facility.

Before starting the various recovery tests, a job was started which repeatedly updated the RACF database. The job executed on various systems and ran a REXX EXEC which processed as follows:

```
do 5000 times
  add a new dataset profile
  permit access to the new profile
  delete the profile
end
```

To prevent reuse of the already defined profiles, a time variable was used as a qualifier in the various profiles.

3.5.1 Test Configuration

The test configuration is shown in Figure 7. RACF 2.1.0 is active on all six systems in the sysplex and operates in RACF sysplex data sharing mode.

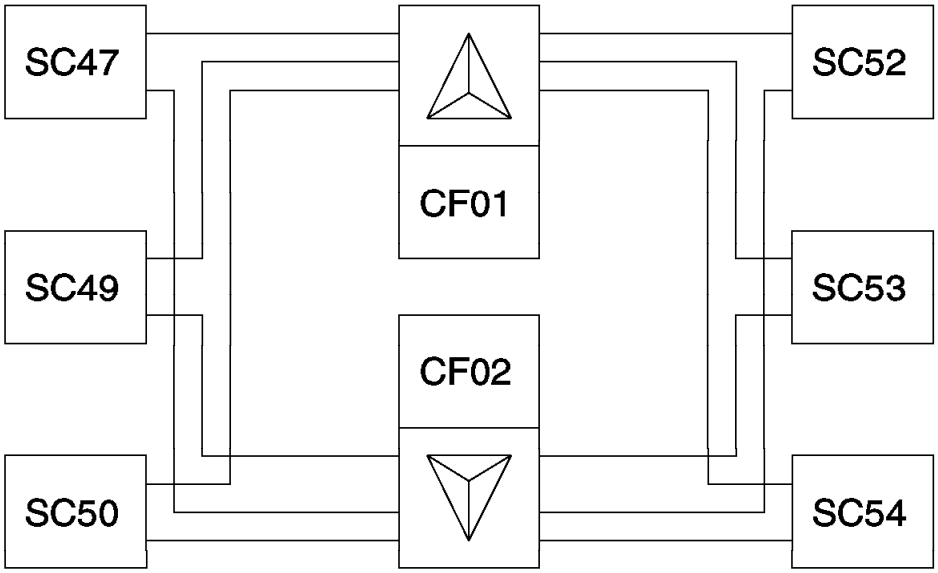

Figure 7. Test Configuration

The following structures are defined in the coupling facilities:

- Coupling Facility CF01 has the following structures:

  STR_JESCKPT1 JES2, structure is ALLOCATED

  IXC_CICS CICS, structure is ALLOCATED

  IRRXCF00_P001 Primary RACF database, structure is ALLOCATED

- Coupling Facility CF02 has the following structures:

  IRRXCF00_B001 Backup RACF database, structure is ALLOCATED

3.5.2 Rebuild a RACF Structure

An attempt was made to move an existing and allocated RACF structure to a coupling facility *other* than the one where the structure originally resided. The following MVS command is used to rebuild structures in the coupling facility:

```
SETXCF START,REBUILD,STRNAME=structure_name,LOCATION=OTHER
```

Notes:

1. RACF support for the “SETXCF START,REBUILD....” command is introduced by APAR OW02202.
2. After APAR OW02202 is installed, all systems in the sysplex must be IPLed because some of the code is only picked up at initialization time of RACF.
3. RACF will *not* support the “SETXCF STOP,REBUILD....” command as well as the “LOCATION=OTHER” parameter on the “SETXCF START,REBUILD....” command.

When a “SETXCF START,REBUILD...” command has been issued through a console, the *coupling facility event exit* will notify RACF about the REBUILD request. As each system in the sysplex will receive the same signal, RACF has no way of knowing which system issued the request and it will internally assign one system as being the coordinator for the REBUILD request. The messages ICH15019I and ICH15020I, which indicate the start and the end of the REBUILD sequence may, therefore, be issued on any console in the sysplex.

RACF will start the REBUILD process the standard way, no matter whether the request was initiated through a coupling facility structure problem or through a “SETXCF START,REBUILD...” request. Internally however, RACF will use the RVARY logic to quiesce I/O operations to the RACF database, then disconnect all members from the structures, allocate new structures and re-establish connections to the new structures.

Regarding the “LOCATION=OTHER” parameter: as the SETXCF command is handled by MVS Cross-system Extended Services (XES), RACF does not know about it’s presence and cannot issue a warning message that the parameter is not supported.

The following MVS command was used to rebuild the cache structure in (another) coupling facility:

```
SETXCF START,REBUILD,STRNAME=IRRXCF00_B001,LOCATION=OTHER
```

Note: Although it was known that it is not supported, the “LOCATION=OTHER” parameter was used. Neither RACF or MVS will tell you that the parameter is not supported. This might lead to confusion.

The following messages were displayed on the various systems:

- System SC52 (the rebuild command was entered on this system)

  ```
  IXC367I THE SETXCF START REBUILD REQUEST FOR STRUCTURE
  IRRXCF00_B001 WAS ACCEPTED.
  ```

- System SC47 (it is unpredictable where these messages appear)

  ```
  ICH15019I INITIATING PROPAGATION OF RVARY COMMAND
            TO MEMBERS OF RACF DATA SHARING GROUP IRRXCF00
            IN RESPONSE TO A REBUILD REQUEST.
  ```

- System SC54 (it is unpredictable where these messages appear)

  ```
  IXC521I REBUILD FOR STRUCTURE IRRXCF_B001 HAS BEEN STOPPED
  ```

- On all the systems in the sysplex that where enabled for data sharing

```
IXC509I CFRM ACTIVE POLICY RECONCILIATION EXIT HAS STARTED.
TRACE THREAD: xxxxxxxx.
IXC509I CFRM ACTIVE POLICY RECONCILIATION EXIT HAS COMPLETED.
TRACE THREAD: xxxxxxxx.

ICH15020I RVARY COMMAND HAS FINISHED PROCESSING.
```

The last message showed that the REBUILD had completed; it could be verified by a MVS command:

```
DISPLAY XCF,STR,STRNM=IRRXCF00_B001
```

This command showed that the structure had been allocated some seconds ago.

Warning: The “LOCATION=OTHER” parameter was not honored (as we had expected); no warning message was issued that the parameter was not supported. In order to rebuild the cache structure in the backup coupling facility, the CRFM policy needs to be updated to reflect the required location of the various cache structures in the available coupling facilities.

The test REXX EXEC that updated the RACF database was ″HALTED″ during the rebuild process. It started automatically after the message:

```
ICH15020I RVARY COMMAND HAS FINISHED PROCESSING.
```

3.5.3 Shutdown of a Coupling Facility

The coupling facility CF02 was disabled through a “SHUTDOWN” command at the Hardware Management console of the 9672 machine. Figure 8 depicts the test configuration after the SHUTDOWN command.

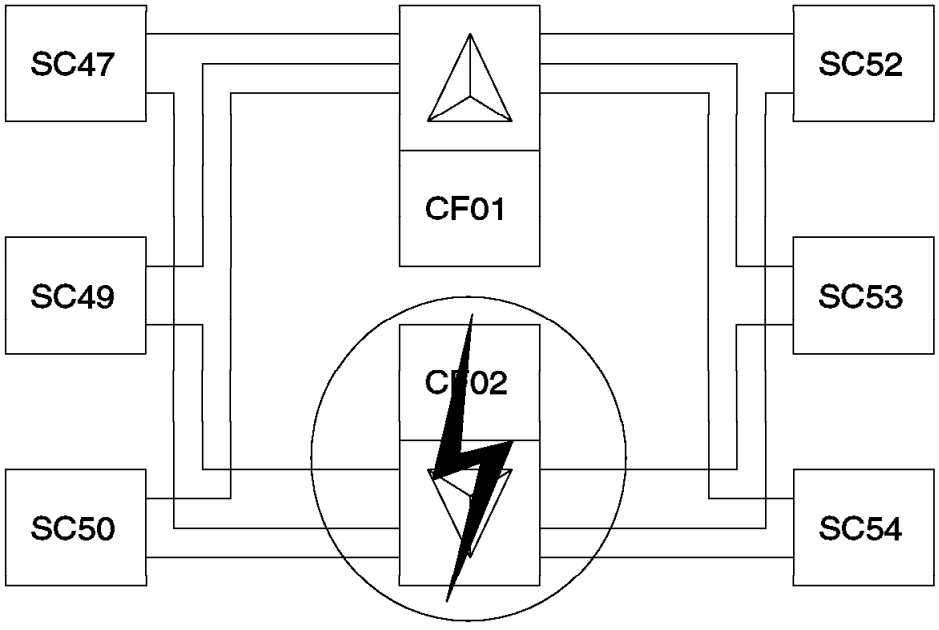

Figure 8. Shutdown of Coupling Facility CF02

This resulted in several messages on each system's console. The messages on system SC52 are shown below:

```
IXL158I PATH 11 IS NOW NOT-OPERATIONAL TO CUID: FFFB
                 COUPLING FACILITY 009672.IBM.02.000000040104
                                   PARTITION: 1  CPCID: 01
IXC518I SYSTEM SC52 NOT USING
        COUPLING FACILITY  009672.IBM.02.000000040104
                           PARTITION: 1    CPCID: 01
        NAMED CF02
       REASON: CONNECTIVITY LOST.
       REASON FLAG: 13300002.

IRRX015A A LINK FAILURE WAS DETECTED BY MEMBER SC52
         FOR STRUCTURE IRRXCF_B001 CORRESPONDING TO RACF
         DATABASE SYS1.RACF.BKUP1

IRRX004I MEMBER SC52 IN READ-ONLY MODE

IXC509I CFRM ACTIVE POLICY RECONCILIATION EXIT HAS STARTED.
TRACE THREAD: xxxxxxxx.

IXC509I CFRM ACTIVE POLICY RECONCILIATION EXIT HAS COMPLETED.
TRACE THREAD: xxxxxxxx.

IRRX000I MEMBER SC52 IS IN DATA SHARING MODE.
```

RACF was able to recover from the loss of a coupling facility as indicated by the message:

```
IRRX0001 MEMBER SC52 IS IN DATA SHARING MODE.
```

The total elapse time of this recovery, with only our small test program running to update the RACF database, was about 5 minutes.

The test REXX EXEC batch job (continuously trying to update the RACF database) caused the following extra messages on the system console (SC47) where the job was running. These messages appeared each time the job was trying to update the RACF database.

```
IRRX016I RACF MEMBER SC47 DETECTED A COUPLING FACILITY 005
         ERROR DURING WRITE
         DATABASE NAME = SYS1.RACF.BKUP1
         XES STRUCTURE NAME = IRRXCF00_B001
         XES TOKEN = X'C9E7C3D3D6F0F0F57FFCB1E80001001D'
         XES LOCAL CACHE INDEX = X'00000019'
         RACF RBA = X'00000000C000'
         XES ERROR CODE = X'0C1C0C06'

IEA794I SVC DUMP HAS CAPTURED: 008
DUMPID=006 REQUESTED BY JOB (JENNISKA)
DUMP TITLE=ICHRST00-RACF SVCS,ABEND CODE=484-080,SVC=IRRRDF00,U
           SER=HAIMO   ,GROUP=SYS1

ICH409I 485-050 ABEND DURING RACDEF  PROCESSING
```

Notes:

1. There is no error recovery at all in the test REXX EXEC program.
2. The RACF ABEND Code 485 indicates that the RACF Manager returned an invalid return code from a “RACROUTE REQUEST=DEFINE” request. The reason code 050 indicates that either the RACF database is locked by a RACF utility (which is not the case) or that an attempt is made to update the RACF database from a system in a sysplex which is in read-only mode (which is the case).
3. The job remained active on system SC47 and resumed normal operations when the system was in data sharing mode again.

A DISPLAY command for the backup RACF database structure shows that the structure for the RACF database has moved to the alternate coupling facility. All systems regained access to it. RACF did resume normal operation and all systems were back in data sharing mode. RACF has recovered successfully.

3.5.4 Power Loss in a Coupling Facility

Instead of the (more or less) “orderly” shutdown of a coupling facility in the previous test, we also tested a more realistic scenario: power loss in a coupling facility.

The same type of error recovery occurred as in the previous test. All systems went into read-only mode, the structure was moved to the alternate coupling facility and RACF resumed operating in RACF sysplex data sharing mode.

3.5.5 Breakdown of a Coupling Facility Link

Figure 9 on page 32 depicts the test configuration for this test. In this test, a coupling facility link error was produced by deactivating a path between the coupling facility and a system. By varying a path off-line from the hardware management console of the 9672 machine, a hardware failure is simulated.

As a result of this error, the system that lost the connectivity entered read-only mode and lost connectivity to the structure for the RACF database.

When the connection was re-established, the system automatically reconnected to the structure and came back into RACF sysplex data sharing mode.

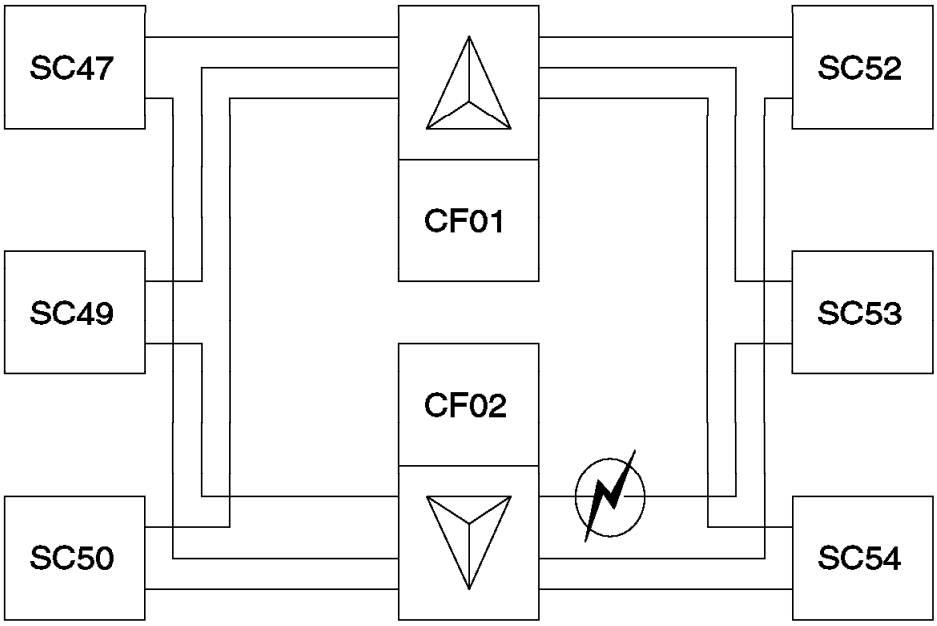

Figure 9. Breakdown of a Coupling Facility Link

3.5.6 Coupling Facility Recovery Matrix

Figure 10 on page 33 depicts the various coupling facility error recovery scenarios. The vertical columns describe the results of the various recovery actions and the conditions that were observed during the error recovery. The horizontal columns describe the various error conditions and the operator initiated recovery scenarios.

In the vertical column you will find answers to the following questions:

- Is the alternate coupling facility used to rebuild the RACF cache structure?
- Is it necessary to update the CFRM policy to force the usage of the alternate coupling facility?
- Is the RACF database available for updates during the recovery process?

	Alternate CF Used	CFRM Policy Update Required	RACF Database available for updates
RVARY NODATASHARE / DATASHARE	YES	YES Note 1	YES Note 4
CF Failure (Power down or shutdown)	YES	NO	NO Note 3
CF Link Failure Note 5	NO	NO	NO
REBUILD (APAR OW02202)	YES Note 2	YES Note 1	YES Note 4

Figure 10. Coupling Facility Recovery Matrix

Notes:

1. The RACF cache structure is rebuilt in the same coupling facility if the CFRM policy is not updated.
2. RACF ignores the LOCATION=OTHER parameter on the SETXCF START,REBUILD command.
3. The systems in the sysplex enter READ ONLY mode during the automatic REBUILD. Updates to the RACF database will fail with a IRRX016I error message that is followed by a ICH409I 485-050 ABEND message. (Error recovery procedures in applications might cause other symptoms).
4. RACF database updates are “halted” until the REBUILD is finished.
5. Systems in the sysplex that experience a LINK failure enter READ ONLY mode. (See also note 3) There is no REBUILD process started.

Chapter 4. RACF Dynamic Exits

This chapter describes a solution that allows the exploitation of the MVS/ESA *dynamic exits facility*. Standard RACF 2.1.0 does not directly support changing of exits dynamically.

System installation exits, such as the RACF exits, offer subsystems and user applications an opportunity to interrupt the system's processing for a number of good reasons; generally the reason is for the system to obtain information on which to base further processing. Often, however, programs take advantage of an installation exit to do processing of their own.

RACF provides a number of installation exits that enable the installation to use their own routines to enhance the facilities offered by RACF, as well as to optimize its usability.

The RACF initialization routine loads the exit routines during system IPL and places the exit addresses in the RACF communication vector table (RCVT). If RACF determines (through a search of the LPA) that the exit routines were not supplied, RACF sets the RCVT fields pointing to the exit routine to zero. If an installation changes an exit, they must re-IPL MVS for the changes to take effect.

In a sysplex environment, you are very interested in avoiding sysplex-wide IPLs. You can IPL one system at a time, but service agreements can be difficult to keep if you have to IPL all systems at the same time.

RACF exits might affect certain authorization checks for specific users or resources. The installation does not have a single image of the RACF environment if different systems in a sysplex operate with different versions of RACF exits. A solution that exploits the dynamic exits facility can avoid this problem. This facility allows the installation to make the same version of a (RACF) exit available and operating at the same time across the systems in a sysplex without the need to IPL the systems.

But remember, dynamic exits might be *a must* for some very valid operational or technical reasons; they can be a *nightmare* for the auditor. The auditor usually keeps a listing of all the current RACF exits since they can change the security process from what is expected according to the profile definitions. If RACF exits are changed dynamically, the auditor might lose control of what exit (and what security process) is currently active.

SAF protection on the various operator commands that are needed to implement this dynamic exit facility can lower the risk of unauthorized or uncontrolled dynamic exit updates.

4.1 Introduction to Dynamic Exits

The dynamic exits facility is a set of services implemented by the following facilities or combination of these facilities:

- The EXIT statement of the PROGxx parmlib member.

  The EXIT statement allows an installation to add exit routines to an exit, delete an exit routine for an exit, change the state of an exit routine, change the attributes of an exit, and undefine an implicitly defined exit.

The PROGxx EXIT statement interacts with the PROG=xx parameter of IEASYSxx and the SET PROG=xx command. At IPL, operators can use PROGxx to specify the particular PROGxx parmlib member the system is to use. During normal processing, operators can use the SET PROG=xx command to set a current PROGxx parmlib member.

See *MVS/ESA SP V5 Initialization and Tuning Reference* for more information about the PROGxx parmlib member.

- The SETPROG EXIT operator command.

  This command performs the same function as the EXIT statement of the PROGxx parmlib member.

  See *MVS/ESA SP V5 System Commands* for information about the SETPROG EXIT command.

- The CSVDYNEX macro.

  Through the CSVDYNEX macro you can define an exit, control its use, and associate exit routines with it. None of these actions require a system IPL.

The solution that is described in this chapter uses a combination of these facilities. The dynamic exits implementation that is described is a very basic form. There is almost no error recovery implemented, and the additional security features of the dynamic exits facility are not used. This implementation should be used as an example to base your own developments.

4.2 Sample Dynamic Exit for ICHRIX01

The sample solution that is described in this chapter consists of the following:

- A *dynamic exit stub* that replaces the standard RACF exit code. This stub calls the real RACF exit (the dynamically loaded or modified exit).

  This sample is using the RACROUTE REQUEST=VERIFY(X) preprocessing exit ICHRIX01.

- The module (LOCRIX01) that contains the real RACF exit. The module must be available in a LNKLST library. This module can be refreshed to reflect the updated exit design.

- A definition in SYS1.PARMLIB(PROGxx) to associate a exit routine with an existing exit. The following definition defines *implicitly* the exit DYN.ICHRIX01 and will load module LOCRIX01 from a library in the linklist concatenation.

```
EXIT ADD
     EXITNAME(DYN.ICHRIX01)
     MODULE(LOCRIX01)
```

  This definition is done before RACF becomes active (during the system IPL). The dynamic exit stub is coded so that even the first RACROUTE REQUEST=VERIFY(X) after the IPL is calling the LOCRIX01 module.

- A CSVDYNEX macro in the dynamic exit stub to call the real exit. The format of the macro is:

```
CSVDYNEX REQUEST=CALL,EXITNAME=LEX,........

LEX   DC   CL16'DYN.ICHRIX01'
```

  The dynamic exit stub verifies whether the exit routine is already defined to the dynamic exit facility.

It uses the following CSVDYNEX macro to obtain this information:

```
      CSVDYNEX REQUEST=QUERY,EXITNAME=LEX,QTYPE=CALL,                *
            RETCODE=LRETCODE,RSNCODE=LRSNCODE,.........

LEX   DC    CL16'DYN.ICHRIX01'
```

The return code and reason code might indicate that the exit is only implicitly defined. What is the case in this example; a SYS1.PARMLIB(PROGxx) EXIT ADD statement defines the exit implicitly.

The dynamic exit stub will now define the exit *explicitly* to the dynamic exit facility by using the following macro:

```
      CSVDYNEX REQUEST=QUERY,EXITNAME=LEX,AMODE=31,PERSIST=IPL,      *
            RETCODE=LRETCODE,RSNCODE=LRSNCODE,.........

LEX   DC    CL16'DYN.ICHRIX01'
```

This define routine is executed only once. The REQUEST=QUERY macro detects at the next execution that the exit is already explicitly defined to the dynamic exit facility. The code skips the define routine in that case.

The process flow for this RACF dynamic exit implementation is depicted in Figure 11.

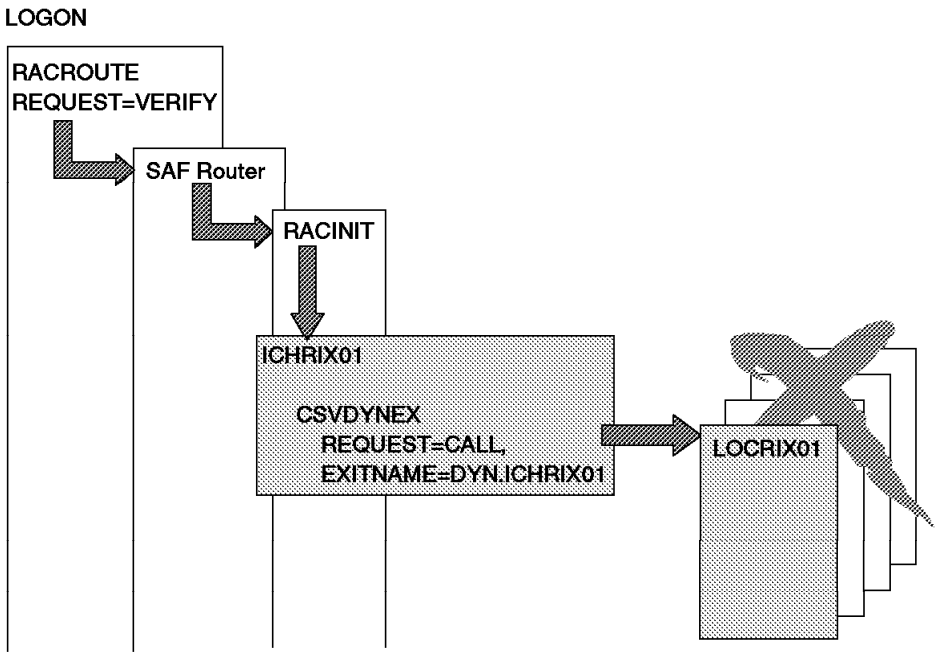

Figure 11. Process Flow of the Dynamic Exit Implementation

To replace a dynamic exit routine, you can either delete the exit routine and add it again, using another name, or change the state of the exit routine to inactive and then change it to active. You can use any of the following methods to replace a dynamic exit routine:

- Use the CSVDYNEX macro to delete the exit routine and add a new one, or make the exit routine inactive and then make it active.

- Use the SETPROG EXIT operator command to delete the exit routine and add a new one.
- Specify a new or modified PROGxx parmlib member that deletes the old exit routine and adds a new one on the MODNAME parameter. Then use the SET PROG= command.
- Modify the PROGxx parmlib member to name a different exit routine on the MODNAME parameter of the EXIT ADD statement. Use the PROG= system parameter with a PROGxx parmlib member at the re-IPL.

In this solution you can use the SETPROG operator command to refresh the RACF exit by performing the following steps:

1. Update the library resident copy of the exit module.
2. Perform a LLA refresh command.
3. Vary the current copy of the exit inactive by using:

   ```
   SETPROG EXIT,MODIFY,
     EXITNAME=DYN.ICHRIX01,MODNAME=LOCRIX01,STATE=INACTIVE
   ```

4. Make the new copy of the exit active by using:

   ```
   SETPROG EXIT,MODIFY,
     EXITNAME=DYN.ICHRIX01,MODNAME=LOCRIX01,STATE=ACTIVE
   ```

Between the inactive and active stage of the exit the CSVDYNEX macro in the dynamic exit stub will fail with a message stating that there is no active exit routine associated with the exit. The stub should provide the appropriate error recovery for this situation. The operator commands can be replaced by the following commands:

```
SETPROG EXIT,DELETE,EXITNAME=DYN.ICHRIX01,MODNAME=LOCRIX01

SETPROG EXIT,ADD,EXITNAME=DYN.ICHRIX01,MODNAME=LOCRIX01,STATE=ACTIVE
```

In a sysplex running MVS 5.1, you could use the new ROUTE command so that the SETPROG command will execute at the same time on all systems in the sysplex.

The CSVDYNEX macro can be used to perform the same type of operation that is executed by using the operator commands. This will reduce the time that there is no exit routine associated with the exit. It will also increase the problems on how to synchronize the exit routine update with the execution of the program that is performing the CSVDYNEX macros with a REQUEST=DELETE and a REQUEST=ADD statement.

See Appendix G, “Dynamic RACF Exit - Sample Code” on page 95 for a sample exit named ICHRIX01 that uses MVS 5.1 Dynamic Exit Facilities.

The solution that is described in this chapter does not work for the RACROUTE REQUEST=FASTAUTH exits, nor for the SAF router exit, nor for the naming convention table.

Warning: The data security monitor (DSMON) produces a RACF exits report. This report lists the names of all the installation-defined RACF exit routines and specifies the size of each exit routine module. If this dynamic exit solution is implemented, the length of the dynamic exit stub is printed, not the length of the exit itself.

Chapter 5. RACLIST Enhancements

In order to get a better understanding of the RACROUTE REQUEST=LIST GLOBAL=YES enhancements, and how they work together with the new RACF sysplex communication facilities, a small Assembler demonstration program, that exploits the new facilities, is developed. This chapter provides a detailed overview of these new facilities. It describes also how this demonstration program exploits these new facilities.

Appendix H, “Source Code for a Demonstration Program” on page 99 provides the source listing of this demonstration program.

5.1 Description of Demonstration Program

The basic purpose of this demonstration program is to show how various updates to RACF profiles are picked up by different RACROUTE calls. These RACROUTE calls do various kinds of authorizations checks, or they are used to retrieve information from a RACF profile.

Profile updates are propagated within a single MVS image by using a new LIST technique in conjunction with a SETROPTS RACLIST REFRESH. This SETROPTS command is propagated across the systems in a sysplex environment by using RACF sysplex communication.

Figure 12 depicts the flow of the demonstration program. The program is testing a profile in the FACILITY class.

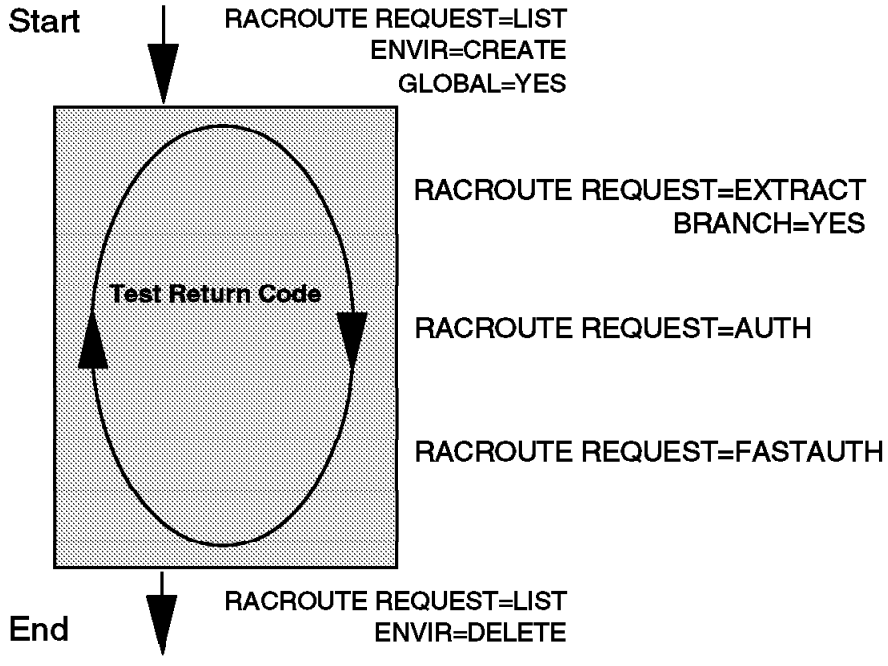

Figure 12. Demonstration Program Flow

The program performs the following actions until a refreshed profile is detected:

1. Depending on a start parameter, the FACILITY class is RACLISTed by the program executing:

   ```
   RACROUTE REQUEST=LIST,ENVIR=CREATE,CLASS=FACILITY,GLOBAL=YES
   ```

2. According to the start-up parameter, different RACROUTE macros are tested:

   a. RACROUTE REQUEST=EXTRACT,BRANCH=YES
   b. RACROUTE REQUEST=FASTAUTH
   c. RACROUTE REQUEST=AUTH

3. When the program does pick up the refreshed profile, it terminates processing. The program relinquish its access to the RACLIST data space by using: RACROUTE REQUEST=LIST,ENVIR=DELETE

5.1.1 Test Scenarios

Different RACROUTE requests are tested with this program. The program has various start parameters to perform these different tests. To control which test should be performed you can enter the start parameter as follows: **REQUEST=**

A REQUEST=AUTH to perform a authorization check for a RACF protected resource, based on a profile in the RACF database.

E REQUEST=EXTRACT,BRANCH=YES to retrieve fields from a RACF profile. General resource profiles that can be loaded into storage are candidates for branch entry EXTRACT. An installation can use the SETROPTS RACLIST command or the RACROUTE REQUEST=LIST,GLOBAL=YES command to create a *global listing of profiles* in a data space.

Note: RACROUTE REQUEST=LIST,GLOBAL=NO creates a listing of profiles in the caller′s address space, but does not create a global listing of the profiles.

F REQUEST=FASTAUTH to check authorization for access to a resource. This macro verifies access to those resources whose RACF profile have been loaded into main storage by the RACROUTE REQUEST=LIST facility.

Dependending on the type of test that is performed, the class FACILITY is RACLISTed, either by a SETROPTS RACLIST(FACILITY) from a TSO/E command line, or by a RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES within the program. The utilization of the REQUEST=LIST parameter is controlled by the program start parameter RACLIST= Y or RACLIST= N.

An additional parameter is required for the REQUEST=EXTRACT test; **OWNER=***UserID*. This test is searching for the specified user ID in the owner field of a specified profile in the FACILITY class.

The REQUEST=AUTH test is checking the UPDATE authority for the submitting user ID for the resource that is protected by the specified profile in the FACILITY class. The REQUEST=FASTAUTH is testing for CONTROL authority to the same resource.

The RACROUTE macro tests are based on the availability of the FACILITY class profiles in a RACLIST data space. To reflect updates to these profiles, you have to refresh the class by using SETROPTS RACLIST(FACILITY) REFRESH. RACROUTE does not provide a facility to refresh profiles that are RACLISTed through the RACROUT REQUEST=LIST,GLOBAL=YES.

5.1.2 Observations

All the findings and conclusions after running this demonstration program where as expected according to the description in the various RACF documents. This topic provides an overview of the highlights. These findings and conclusions can also be arranged by combining various chapters in different RACF documents.

- The tests are performed with RACF V1R9 and with RACF V2R1. The tests shows different results, since RACF V1R9 does not support the GLOBAL=YES parameter in the RACROUTE REQUEST=LIST macro, and RACF V1R9 does not support RACF sysplex communication.

  RACF V1R9 ignores the GLOBAL=YES parameter in the RACROUTE REQUEST=LIST. This will create a listing of profiles in the user's address space, but does not create a *global listing of profiles*. For example, following RACROUTE REQUEST=EXTRACT,BRANCH=YES macros are failing with the following error code:

  ```
  SAF Return Code 04  RACF Return Code 08  Reason Code 0C

  The requested function could not be performed.
  Class not RACLISTed (branch EXTRACT).
  ```

- The RACROUTE REQUEST=FASTAUTH is not using profiles that are loaded into a data space by using SETROPTS RACLIST. This macro is using either profiles that are loaded into the application address space by using RACROUTE REQUEST=LIST (this is not part of the demonstration program), or profiles that are loaded into a RACLIST data space through a RACROUTE REQUEST=LIST,GLOBAL=YES.

  If the RACROUTE REQUEST=FASTAUTH is executed in an application that did not execute a RACROUTE REQUEST=LIST prior to the authorization check, it will fail with the error message:

  ```
  SAF Return Code 04 RACF  Return Code 04

  The requested function could not be performed.  The resource or class name
  is not defined to RACF or the class has not been RACLISTed.
  ```

  This error code will appear even if the class is RACLISTed through a RACROUTE REQUEST=LIST by another application.

- Profile updates for RACLISTed profiles are only active after a refresh operation of the RACLIST data space. Even if the administrator or end user receives the message: "YOUR ACCESS GIVEN" with the requested access, or a LIST command for the profiles shows the updated profile, RACLISTed profiles need to refreshed to become active.

  The administrator will also not always be informed that the profiles are RACLISTed. Only in case of a SETROPTS RACLIST will he get the message:

  ICH10006I RACLISTED PROFILES FOR *class_name* WILL NOT REFLECT THE ADDITION(S) UNTIL A SETROPTS REFRESH IS ISSUED.

- The RACROUTE REQUEST=AUTH test did always pick up the changed profiles since this macro is using either the RACLISTed profiles after a REFRESH operation, or from the resident data blocks. in case the profiles are not RACLISTed.

5.1.3 Create the RACLIST Data Space

The RACLIST data space is created through one of the following facilities:

- SETROPTS RACLIST(*class_name*)
- After installing RACF V1R2 by using RACROUTE REQUEST=LIST, GLOBAL=YES

The demonstration program is designed to perform the various test scenarios according to start-up parameters. Therefore, it is possible to run the test with or without RACF profiles explicitly RACLISTed in a data space. When the LIST function of the RACROUTE macro is required, the following macro is executed:

```
EXECLIST RACROUTE REQUEST=LIST,                                    *
               ENVIR=CREATE,                                       *
               RELEASE=2.1,                                        *
               GLOBAL=YES,                                         *
               CLASS=NAMFAC,                                       *
               WORKA=(R5),                                         *
               MF=S

NAMFAC   DC    CL8'FACILITY'
```

When the profiles are RACLISTed through REQUEST=LIST,GLOBAL=YES, they can be refreshed through the RACF command:

```
SETROPTS RACLIST(FACILITY) REFRESH
```

Figure 13 depicts how this environment is created.

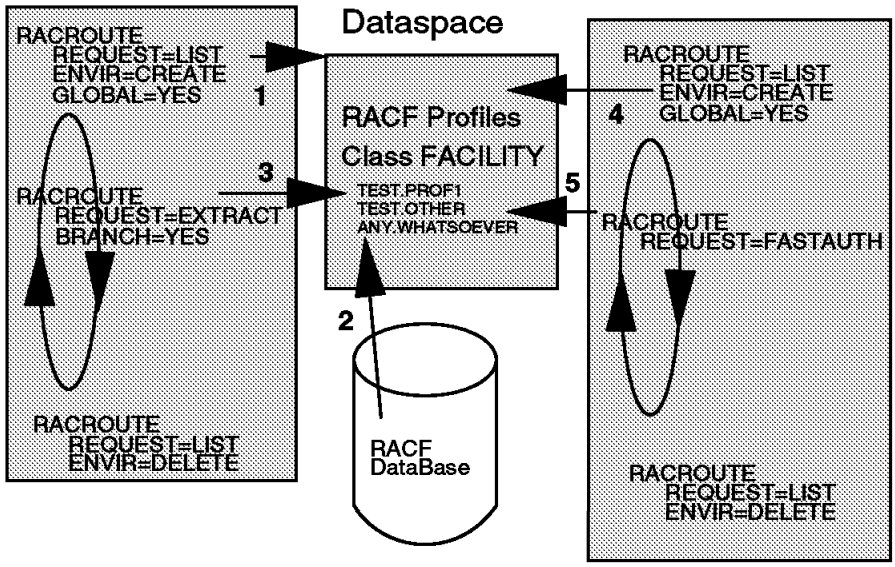

Figure 13. Data Space Creation and Usages

The following steps occur:

1. The RACLIST data space is created by a program executing the REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES macro. If the RACLIST data space exists already, created either by another program or through a SETROPTS RACLIST(FACILITY) command, the new program gets a pointer to this data space.
2. After the RACLIST data space is created, the RACF profiles of the specified class are loaded into the RACLIST data space.

3. After loading the profiles, the program can use these profiles with the subsequent RACROUTE calls.
4. If a second copy of the demonstration program is started on the same MVS image and operating on the same RACF class, it will get a pointer to the already existing RACLIST data space and will not create a new RACLIST data space.
5. RACROUTE calls in this second copy of the demonstration program use the profiles in the already existing data space.

Although it does not make any difference for the operation, it is recommended that you SETROPTS RACLIST the classes that are are RACLISTed through REQUEST=LIST,GLOBAL=YES and for which RACLIST=ALLOWED is specified in the class descriptor table. It will make administration somewhat simpler and will cause less confusion.

By updating a profile of a SETROPTS RACLISTed class, the administrator is informed that "RACLISTed profiles for class *class_name* will not reflect the addition(s) until a SETROPTS RACLIST(*class_name*) REFRESH is issued." The administrator does not get this message if the class is only loaded into the RACLIST data space by using the RACROUTE macro.

RACF shows the RACLISTed classes on the SETROPTS LIST command as follows:

```
.........
.........
SETR RACLIST CLASSES = TSOPROC ACCTNUM TSOAUTH OPERCMDS PTKTDATA PTKTVAL
GLOBAL=YES RACLIST ONLY = FACILITY
.........
.........
```

The demonstration program is assembled with the RACF 2.1 macros. If it runs on a system with a RACF release earlier than RACF 2.1 or when RACF 2.1 is run on MVS/ESA SP 4.2.0, the parameter GLOBAL=YES is ignored. In this case the RACF profiles are RACLISTed in local private storage, and cannot be refreshed with the SETROPTS RACLIST(FACILITY) REFRESH command.

After starting the programs, a SETROPTS RACLIST(FACILITY) REFRESH is performed to propagate an updated profile. The demonstration programs loop until they detect the updated profile. The refresh scenario is depicted in Figure 14 on page 44.

Before the program terminates, the following macro is executed to relinquish the address space from the RACLIST data space:

```
EXECDEL  RACROUTE REQUEST=LIST,                                        *
               ENVIR=DELETE,                                           *
               RELEASE=2.1,                                            *
               CLASS=NAMFAC,                                           *
               WORKA=(R5),                                             *
               MF=S

NAMFAC   DC    CL8'FACILITY'
```

5.1.4 Refresh the RACLIST Data Space

After updates to RACLISTed profiles, the RACLIST data space that contains these profiles must be refreshed to reflect these profile updates. Figure 14 depicts the refresh operation of the RACLIST data space.

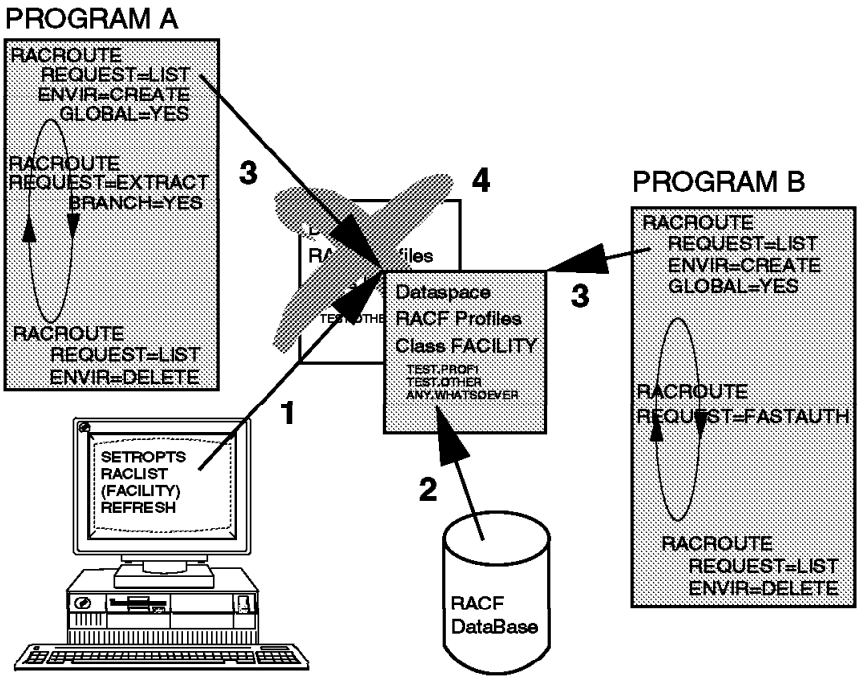

Figure 14. Refresh of the RACLIST Data Space

The flow of this refresh operation is as follows:

1. From the TSO/E command line the following RACF command is issued:

   ```
   SETROPTS RACLIST(FACILITY) REFRESH
   ```

2. The SETROPTS command creates a new RACLIST data space. The updated RACF profiles of the particular class (FACILITY) are loaded into that RACLIST data space.
3. The pointer to the RACLIST data space that is kept by the running programs is switched to the new RACLIST data space.
4. When all running programs address the new RACLIST data space, the old RACLIST data space gets deleted.

There are some comments to make on the refresh process. In topic 5.1.3, “Create the RACLIST Data Space” it is already explained that the user that is updating a profile of a RACLISTed class is not always informed of the fact that this class is RACLISTed. Only if the class is RACLISTed by using SETROPTS RACLIST(*class_name*) will he get that message.

Even when he gets the message ″YOUR ACCESS GIVEN″ indicates the requested value, or a LIST of that particular profile shows the right accesses after a profile update, his changes might still not be effected. The running programs are still using the old RACLIST data space with the old profiles. A refresh command is needed to get the new profiles to be used by the running applications.

Even applications that are starting after the profile updates, and perform a RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES, are using the old profiles since they will get a pointer to the already existing data space. The RACLIST data space also stays active after all the address spaces (applications) that worked with it are finished. This is independent whether the RACROUTE REQUEST=LIST,ENVIR=DELETE macro is executed to relinquish the address space from the RACLIST data space.

5.1.5 Refresh the RACLIST Data Space in a Sysplex

When RACF is enabled for sysplex communication, RACF propagates a SETROPTS RACLIST (*class_name*) or SETROPTS RACLIST (*class_name*) REFRESH command issued from any one system to the other systems in the data sharing group if the command is successful on the system on which it was entered. RACF will call the system that is used to issue that command the "coordinator."

If a refresh is being done, RACF continues to use the old in-storage profiles for authorization requests until the new ones are created. When all systems have completed rebuilding the local RACLIST data spaces, the coordinator signals the members of the data sharing group to discard the old ones, and to begin using the new RACLIST data spaces.

Figure 15 depicts the refresh operation of the RACLIST data space in a sysplex environment.

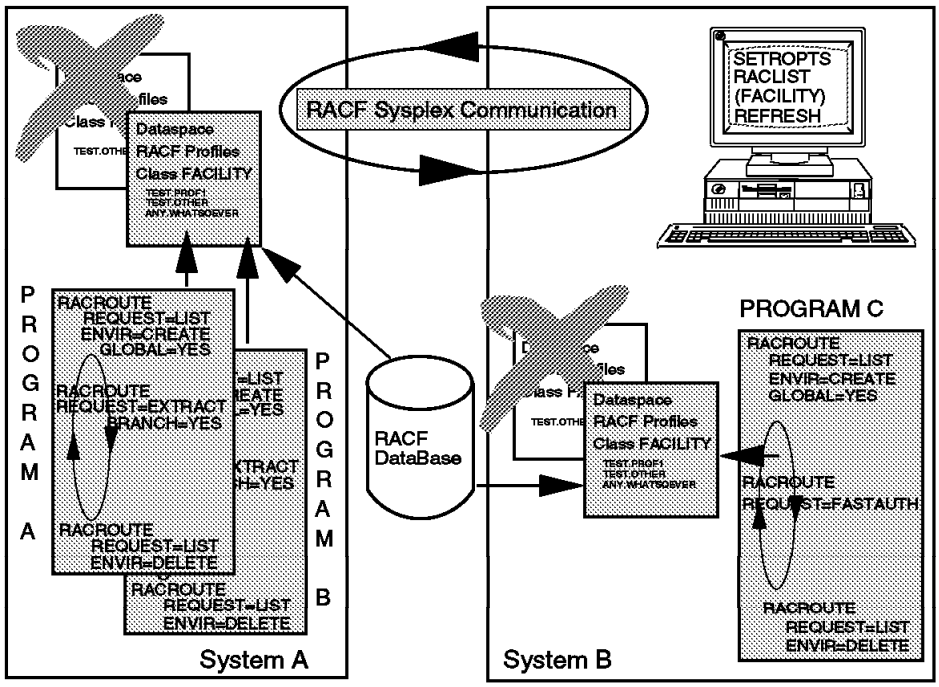

Figure 15. Refresh of the RACLIST Data Space in a Sysplex

If RACF is not enabled for sysplex communication, the administrator must issue the SETROPTS RACLIST(*class_name*) command and the SETROPTS RACLIST(*class_name*) REFRESH command on each system sharing the RACF database.

5.1.6 Delete the RACLIST Data Space

Once created, the RACLIST data space can be refreshed when appropriate by a SETROPTS RACLIST(*class_name*) REFRESH command.

The SETROPTS NORACLIST (*class_name*) command can be used to delete the RACLIST data space. This command should not be issued until all of the applications that accessed the RACLIST data space by issuing a RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES have relinquished their access by issuing RACROUTE REQUEST=LIST,ENVIR=DELETE requests.

If the RACLIST data space is deleted while there is still a copy of the demonstration program running that need these profiles for a RACROUTE REQUEST=FASTAUTH, that copy will fail with an error message:

```
SAF Return Code 04  RACF Return Code 04

The requested function could not be performed.  The class was RACLISTed
by RACROUTE REQUEST=LIST,GLOBAL=YES, but the data space has been deleted.
```

When RACF is enabled for sysplex communication, RACF propagates the SETROPTS NORACLIST command to other systems in the data sharing group if the command was successful on the system in which it was entered. If RACF is not enabled for sysplex communication, the administrator must issue the SETROPTS NORACLIST command on each system sharing the RACF database.

Note: As with the SETROPTS RACLIST REFRESH, the SETROPTS NORACLIST command processes not only the class specified on the command, but all valid classes sharing the same POSIT.

5.1.7 Sample RACROUTE Macros Used in the Demonstration Program

The following topics provide some samples of the RACROUTE macros that are implemented in the demonstration program.

5.1.7.1 RACROUTE REQUEST=AUTH

The RACROUTE REQUEST=AUTH macro checks a user's authority to access a resource, based on a profile in the RACF database when a user requests access to a RACF-protected resource. In the demonstration program the following parameters are specified:

```
EXECAUTH RACROUTE REQUEST=AUTH,                                        *
               RELEASE=2.1,                                            *
               CLASS=CLFAC,                                            *
               ENTITY=(ENTITY),                                        *
               STATUS=ACCESS,                                          *
               MSGSUPP=YES,                                            *
               WORKA=(R5),                                             *
               MF=S

CLFAC    DC    XL1'08'
NAMFAC   DC    CL8'FACILITY'
ENTITY   DS    CL39
```

An installation can optimize RACF performance by carefully deciding to use a RACLIST facility for various RACF classes. The RACLIST operand on the SETROPTS command improves performance by copying generic and discrete

profiles for the designated general-resource class (and each RACLISTable class that shares the same POSIT value) from the RACF database into a data space.

RACROUTE REQUEST=LIST,GLOBAL=YES can also improve performance since it stores the profiles for the requested class also in a data space. This RACLIST data space can be shared by other applications that issue the same request. Additional RACROUTE requests do not access the RACF database; they access the RACLIST data space built by the first RACROUTE REQUEST=LIST,GLOBAL=YES or by the SETROPTS RACLIST command.

If both facilities, SETROPTS RACLIST(*class_name*) and RACROUTE REQUEST=LIST,GLOBAL=YES, are activated for a particular RACF class, they will share the same data space to load the profiles from that class. The demonstration program showed that the in-storage profiles are accessible by the RACROUTE REQUEST=AUTH and can be used for authorization checking. A refresh operation is needed to activate update profiles if they are listed into a RACLIST data space.

If the profiles are not listed into a RACLIST data space, the RACROUTE REQUEST=AUTH is using the profiles in the resident data blocks or, if they are not loaded into these local RACF buffers, it will read the profiles from the cache structures in the coupling facility or direct from RACF database. It is not needed to perform any type of refresh operation to pick up the updated profiles. Every profile update is detected by the normal profile validation facilities, as described in 1.4, “RACF Data Buffer Synchronization” on page 5.

The parameters that are specified in this example are explained as follows:

STATUS=ACCESS
: This request returns the user’s highest current access to the resource specified. Upon successful completion, the user’s access is returned in the RACF reason code, as indicated in the following message:

```
SAF Return Code 00  RACF Return Code 14  Reason Code user's_highest_access

RACROUTE REQUEST=AUTH completed successfully.
```

: In the demonstration program, the reason code is a pointer into a branch table that is used to build a more meaningful message to the operator.

MSGSUPP=YES
: This parameter results in suppressing the WTO message from RACF processing. This parameter affects only message ICH408I and associated auditing support information messages (IRR series) as well as some ICH7000xx messages.

CLASS=*class_name*
: Points to a 1-byte field indicating the length of the class name, followed by the class name.

ENTITY=*resource_name_address*
: Points to an address that contains the name of the profile that is used in this authorization check.

WORKA=*work_area_address*
: This is a required parameter for the execution of a RACROUTE macro.

5.1.7.2 RACROUTE REQUEST=EXTRACT

The RACROUTE REQUEST=EXTRACT macro is used to retrieve or replace certain specified fields from a RACF profile. This macro supports an SRB-compatible branch entry when BRANCH=YES is specified together with the TYPE=EXTRACT parameter. In the demonstration program, the following parameters are specified:

```
EXECEXTR RACROUTE REQUEST=EXTRACT,                                     *
               TYPE=EXTRACT,                                           *
               RELEASE=2.1,                                            *
               BRANCH=YES,                                             *
               SUBPOOL=2,                                              *
               CLASS=NAMFAC,                                           *
               ENTITY=ENTITY,                                          *
               FIELDS=FLDSFAC,                                         *
               WORKA=(R5),                                             *
               MF=S

NAMFAC   DC    CL8'FACILITY'
ENTITY   DS    CL39
FLDSFAC  DC    AL4(2)
         DC    CL8'OWNER'
         DC    CL8'UACC'
```

BRANCH=YES together with TYPE=EXTRACT requires that the specified resource profile is available in a *global listing of profiles* in a RACLIST data space.

An installation can use the SETROPTS RACLIST command or the RACROUTE REQUEST=LIST,GLOBAL=YES command to create a global listing of profiles in a RACLIST dataspace. The RACROUTE REQUEST=LIST,GLOBAL=NO command creates a listing of profiles in the caller's address space, but does not create a global listing of profiles.

When the class is not RACLISTed, neither by the SETROPTS command nor with the LIST function of the RACF RACROUTE macro in the program, the EXTRACT request returns with the following message:

```
SAF Return Code 4  RACF Return Code 8  Reason Code 0C

The requested function could not be performed.  Class not RACLISTED
(branch EXTRACT)
```

Upon return, register 1 contains the address of a result area that begins with a fullword containing the length and subpool number of the area. (It is the responsibility of the application to issue a FREEMAIN to release the area after it uses the result of the macro.)

The parameters that are specified in this example are explained as follows:

TYPE=EXTRACT
: Requests to extract information from any field in any profile, unless BRANCH=YES is specified. (See also BRANCH=YES). If EXTRACT is specified, RACF extracts information from the profile determined by the ENTITY and CLASS keywords.

BRANCH=YES
: Specifies that general resource profiles that can be brought into storage are candidates for branch entry EXTRACT.

For a detailed overview of the differences between BRANCH=YES and BRANCH=NO, refer to the *External Security Interface (RACROUTE) Macro Reference for MVS*.

SUBPOOL=*subpool_number*
Specifies the storage subpool from which the extract-function routine obtains an area needed for the extraction.

CLASS=*class_name*
Specifies the class the entity is in.

ENTITY=*profile_name_address*
Specifies the address of a resource name for which profile data is to be extracted. This area has fixed lengths for certain types of profiles. For example, the area is eight bytes long for a USER or GROUP profile, and 44 bytes long for a DATASET profile. For other profile names, for example for profiles in the FACILITY class, the lengths are determined by the class-descriptor table. The name must be left-justified in the field and padded with blanks.

FIELDS=*field_address*
Specifies the address of a variable-length list. The first field is a four byte field that contains the number of profile fields in the list that follows. Each profile field name is eight bytes long, left-justified, and padded to the right with blanks. Refer to the *External Security Interface (RACROUTE) Macro Reference for MVS* for a detailed list of allowable field names for each type of profile.

WORKA=*work_area_address*
This is a required parameter for the execution of a RACROUTE macro.

MF=S
Indicates the standard form of the macro. By design, the standard form of the macro generates an inline parameter list and then modifies it. The standard form of the macro does three things: It reserves storage, fills in the parameters you have specified on the parameter list, and generates a call to the service routine.

There are four different forms of the RACF macros: standard (S), list (L), execute (E), and modify (M).

5.1.7.3 RACROUTE REQUEST=FASTAUTH

The RACROUTE REQUEST=FASTAUTH macro is used to check a user′s authorization for access to a resource. The difference between this macro and the RACROUTE REQUEST=AUTH macro is that RACROUTE REQUEST=FASTAUTH verifies access to those resources whose RACF profiles have been brought into main storage by the RACROUTE REQUEST=LIST,ENVIR=CREATE (GLOBAL=YES or GLOBAL=NO) facility.

RACROUTE REQUEST=AUTH uses both RACLISTed profiles and non-RACLISTed profiles.

RACROUTE REQUEST=FASTAUTH does not gather statistics or issue SVCs. Therefore, use of this macro is recommended only for applications that have stringent performance requirements.

In order to get the right pointer to the RACLIST data space, it is necessary to perform a RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES prior to the execution of the RACROUTE REQUEST=FASTAUTH macro.

RACROUTE REQUEST=FASTAUTH will not not use the RACLISTed profiles in case the profiles are RACLISTed through one the following operations and there is no RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES in the calling program:

- SETROPTS RACLIST command
- Another application performing a RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES

A RACROUTE REQUEST=LIST,GLOBAL=NO macro loads the profiles in the private storage of the application. The RACROUTE REQUEST=FASTAUTH will always use these profiles.

Several IBM applications use this new form of RACROUTE REQUEST=FASTAUTH processing. Among them are CICS/ESA and OPC/A.

In the demonstration program, the following parameters are specified:

```
EXECFAST RACROUTE REQUEST=FASTAUTH,                                    *
               RELEASE=2.1,                                            *
               CLASS=NAMFAC,                                           *
               ENTITY=ENTITY,                                          *
               ATTR=CONTROL,                                           *
               WORKA=(R5),                                             *
               WKAREA=(R6),                                            *
               MF=S

NAMFAC   DC    CL8'FACILITY'
ENTITY   DS    CL39
```

The parameters that are specified in this example are explained as follows:

CLASS=*class_name*
: Specifies the class the entity is in. If an address is specified, the address must point to an eight byte field containing the class name.

ENTITY=*profile_name_address*
: Specifies the address of a resource name for which profile data is to be extracted. This area has fixed lengths for certain types of profiles. For example, the area is six bytes long for a volume serial number for CLASS=DASDVOL or CLASS=TAPEVOL. For other profile names, for example for profiles in the FACILITY class, the lengths are determined by the class-descriptor table. The name must be left-justified in the field and padded with blanks.

ATTR=CONTROL
: Specifies the access authority the user must have to the resource profile.

WORKA=*work_area_address*
: This is a required parameter for the execution of a RACROUTE macro.

WKAREA
: Specifies the address of a 16-word work area to be used by the macro. This area contains various results after the execution of the macro. For example, it holds the reason code and return code, and sometimes the address of the in-storage profile used to determine the user's authorization.

MF=S

Indicates the standard form of the macro. By design, the standard form of the macro generates an inline parameter list and then modifies it. The standard form of the macro does three things: It reserves storage, fills in the parameters you have specified on the parameter list, and generates a call to the service routine.

There are four different forms of the RACF macros: standard (S), list (L), execute (E), and modify (M).

5.1.8 The RACGLIST Class

RACF uses the RACGLIST class to save the results of in-storage profiles RACLISTed to a RACLIST data space from any of the following commands:

- SETROPTS RACLIST(*class_name*)
- SETROPTS RACLIST(*class_name*) REFRESH
- RACROUTE REQUEST=LIST,GLOBAL=YES

RACF uses the results from the RACLIST data space to create or replace profiles named *class_name_nnnnn* in the RACGLIST class. (Where *nnnnn* begins at 00001).

To enable RACF to use the RACGLIST class, you need to activate the RACGLIST class by using SETROPTS CLASSACT(RACGLIST), and prime RACGLIST for a specific class by issuing the RDEFINE RACGLIST *class_name* command. For example: if the RACGLIST class name profile is FACILITY, RACF creates additional profiles named:

```
FACILITY_00001
FACILITY_00002
FACILITY_00003
```

Figure 16 depicts the flow of RACGLIST processing.

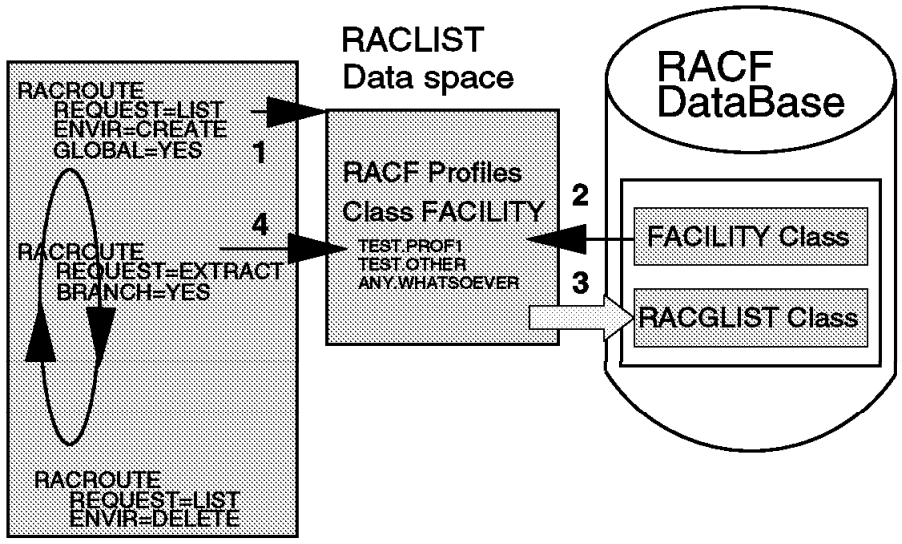

Figure 16. RACGLIST Processing

The flow of the RACGLIST processing is as follows:

1. The RACLIST data space is created by a program executing the REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES macro.
2. After the RACLIST data space is created, the RACF profiles of the specified class (FACILITY) are loaded into the RACLIST data space.
3. If the RACGLIST class is activated and the correct profiles are available in the RACGLIST class, the contents of the RACLIST data space are loaded into the profiles in the RACGLIST class.
4. After loading the profiles into the RACGLIST class, the application that executed the RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES can use the profiles in the RACLIST data space for all subsequent RACROUTE calls.

If a second copy of the demonstration program is started on the same MVS image and operating on the same RACF class, it will get a pointer to the already existing data space and will not create a new RACLIST data space. RACROUTE calls in this second copy of the demonstration program use the profiles in the already existing RACLIST data space.

In a normal RACLIST operation, either by SETROPTS RACLIST or RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES, a subsequent RACLIST request on another system that is sharing the same RACF database might pick up updated profiles since it is using the profiles in the RACF database to build the RACLIST data space. In case of using the RACGLIST class, each system that is sharing the same database loads the profiles from the RACGLIST class into the RACLIST data space. This guarantees, for example, that each system in a sysplex is using the same profiles for authorization checks.

Note: RACGLIST provides a single-system image for security when the installation is using RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES or SETROPTS RACLIST(*class_name*) on multiple systems in a sysplex.

RACF uses these RACGLIST profiles to build the RACLIST data space in any of the following cases:

- For SETROPTS RACLIST(*class_name*) processing during a system IPL
- After a system IPL by RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES processing
- During the processing of a propagated SETROPTS RACLIST or SETROPTS RACLIST(*class_name*) REFRESH command.

This implies that RACF profile updates for classes that are RACLISTed and RACGLISTed are not picked up by these RACLIST requests, even after a system IPL on that particular MVS image. In order to get the applications to use the updated profiles, a refresh operation is required.

Depending on the installation's RACF database set up, it may take less processing time and less I/O to read the stored RACLIST results from the RACGLIST profiles than to retrieve the original discrete and generic profiles from the RACF database for the class name. RACF profiles are usually not in any sequence in the RACF database. They are stored in data blocks that are available all over the database. Index searches are needed to retrieve the profiles.

In a RACGLIST class, the profiles are stored in a structure like they are finally stored in the RACLIST data space. It is no longer necessary to build the new RACLIST data space structure again. Profiles are also read from the RACGLIST class into the RACLIST data space in big chunks of data instead of one profile at the time, like in case of the profiles being read from the database one at the time.

5.1.9 Refresh the RACGLIST Class

When a security administrator needs to make several updates to profiles in a RACLISTed class, systems continue to access the stored profiles until the administrator completes the changes and tells RACF to refresh the profiles (in all the sharing systems) by issuing SETROPTS RACLIST(*class_name*) REFRESH.

The flow of the refresh operation is as follows:

1. From the TSO/E command line the following RACF command is issued:

   ```
   SETROPTS RACLIST(FACILITY) REFRESH
   ```

2. The SETROPTS command creates a new RACLIST data space. The updated RACF profiles of the particular class (FACILITY) are loaded into that RACLIST data space.
3. RACF uses the RACLIST data space information to create or refresh the RACGLIST profiles.
4. The pointer to the RACLIST data space that is kept by the running programs is switched to the new RACLIST data space.
5. When all running programs address the new RACLIST data space, the old RACLIST data space gets deleted.

Installations that are using RACGLIST classes should see decreased accesses to the RACF database when performing sysplex-wide SETROPTS RACLIST REFRESH operations or during initialization when a initial RACLIST is performed.

The RDELETE RACGLIST *class_name* profile will delete the core RACGLIST profile, plus *class_name* profiles, if any, that are associated with it. This will leave the associated RACLIST data space intact.

Warning: Issuing the SETROPTS NORACLIST will delete the RACLIST data spaces on all members of the RACF sysplex data sharing group as well as the *class_name_nnnnn* profiles on the database.

Chapter 6. RACF OpenEdition MVS Support

OpenEdition MVS combines the personal power of the workstation, the flexibility of open systems, and the strength of MVS into a new environment. Offering new, open interfaces for applications and interactive users, OpenEdition MVS supports and fosters a super environment of larger operating systems or servers and of distributed systems and workstations that share common interfaces.

This chapter gives a very brief overview on how to manage the system and data security in a OpenEdition MVS environment. It also provides a brief overview of OpenEdition MVS.

6.1 Introduction to OpenEdition MVS

OpenEdition MVS extends support to the set of standards being developed as the *Portable Operating System Interface* (POSIX). Some of these standards are based on a set of existing functions that were common across several divergent UNIX-based systems.

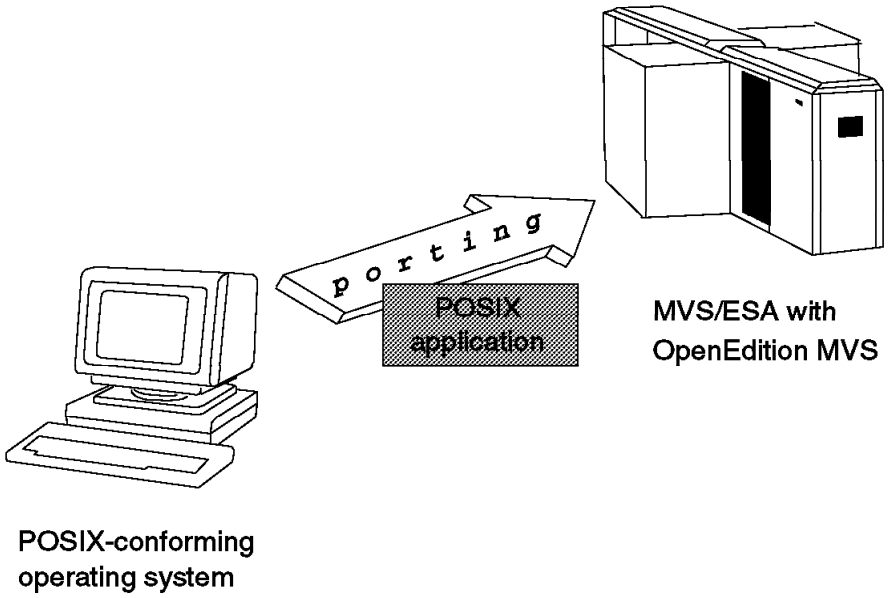

Figure 17. Introducing OpenEdition MVS

Applications that are created conforming to the POSIX standard can be moved, with relative ease, among conforming computer systems and can meet national standards and guidelines.

Some of the benefits of using OpenEdition MVS are:

- Application developers and interactive users have the underlying resources and power of the MVS system available without needing to understand MVS′s own proprietary interfaces.

- Application developers can develop most of their applications at their workstations and then use standard POSIX interfaces to compile, test, and debug their programs at a MVS system.
- Users can exploit the MVS proprietary interface for capabilities not provided by standard interfaces.
- Interactive users can quickly toggle from the POSIX shell to the TSO/E interface to request services not available in the POSIX interface, and then toggle back to the POSIX shell.
- MVS commands are provided that enables the user to copy MVS data sets so they can be accessed by POSIX applications.

Today′s MVS offers large-scale, high-performance transaction, interactive, and batch computing on a scale unmatched by smaller and less sophisticated operating systems. Moreover, MVS offers tools for managing, analyzing, controlling, and optimizing applications and data.

The *open* world promises the economies of standards: strategic skills and knowledge investment, portability of applications and data in a *multivendor* network. It also intends to provide the ability of users and applications to interoperate among systems from different manufactures with a single, common interface.

The MVS component that supports OpenEdition MVS enables two *open systems* interfaces on the MVS operating system: an application program interface (API) and, optionally, an interactive *shell* interface.

With the API, the programs can run in any environment, including batch jobs, in jobs submitted by TSO/E interactive users, and in started tasks, or in any other MVS application task environment. The program can request only MVS services, only OpenEdition MVS services, or both MVS and OpenEdition MVS services.

The optional shell interface is an execution environment analogous to TSO/E, with a programming language of shell commands analogous to the REXX language. The shell work consists of:

- Programs run interactively by shell users
- Shell commands and scripts run interactively by shell users
- Shell commands and scripts run as batch jobs

To invoke the shell, a user logs on to a TSO/E session and enters the TSO/E OMVS command. The OpenEdition MVS environment has other TSO/E commands, for example:

- Logically mount and unmount file systems
- Create directories in a file system
- Copy files to and from MVS data sets

Interactive users can switch from the shell to their TSO/E session, enter commands or do editing, and switch back to the shell.

6.2 Interoperability of OpenEdition MVS

There is a high degree of interoperability between MVS, through the TSO/E facilities, and the OpenEdition shell. Once the *OpenEdition Shell and Utility* feature is installed, users can:

- Request OpenEdition services from the system through shell commands
- Write shell scripts to run tasks (Shell scripts are analogous to REXX EXECs)
- Run OpenEdition programs in the foreground or background

Users of OpenEdition MVS services use similar interfaces on other UNIX based systems, and use terminology different from MVS and TSO/E terminology. To help you understand these differences, a comparison between the environmentals for OpenEdition and TSO/E is provided in 6.8, “Parallels between the TSO/E Environment and the Shell Environment” on page 77.

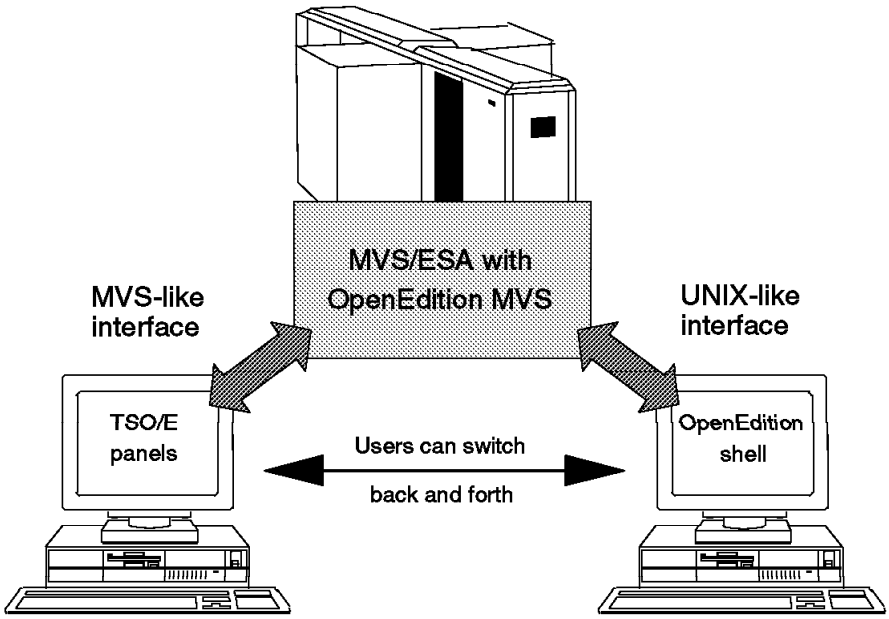

Figure 18. Interoperability of OpenEdition MVS

To begin a shell session, you first log on to TSO/E and then invoke the shell with the TSO/E OMVS command at the TSO/E READY prompt. During a shell session, you can switch temporarily to TSO/E command mode to perform some tasks, without interrupting the shell session. Alternatively, you can enter TSO/E commands on the shell command line and use the TSO/E function key to run the commands.

The interoperability of the OpenEdition MVS allows a user to:

- Move data between MVS data sets and the OpenEdition file system. A user can copy or move MVS data sets into the file system; likewise, he can copy or move file system data into MVS data sets.
- To work with the OpenEdition file system, a user can use:
  - TSO/E commands
  - The panel interface of the OpenEdition ISPF shell
  - Shell commands

- To edit OpenEdition files, a user can use the ISPF/PDF full-screen editor or, in the shell, he can use a shell command to edit the file.

The OpenEdition shell provides commands and utilities. POSIX distinguishes between a command (a directive to a shell to perform a specific task) and a utility (the name of a program callable by name from a shell). Shell commands often have *options* (also known as flags) that a user can specify, and they usually take an *argument*, such as the name of a file or directory.

6.3 Controlling OpenEdition MVS Security

Resource Access Control Facility (RACF) can be used to manage system and data security in a OpenEdition MVS environment by verifying a user and verifying that a user can access a *process* (a program that uses OpenEdition MVS services) or *file*. A user is identified by an *OMVS user ID (UID)*, which is kept in the RACF user profile, and a *OMVS group ID (GID)*, which is kept in the RACF group profile.

The system verifies the user IDs and passwords of the users when they log on to a TSO/E session or when a job starts. When a user in a TSO/E session invokes the shell, RACF verifies that the interactive user is defined to the OpenEdition MVS before the system initializes the shell. When a program requests a service from OpenEdition MVS for the first time, RACF verifies that the user running the program is defined to OpenEdition MVS before the system provides the service.

6.4 Adding Users and Groups for OpenEdition MVS

This section describes how to add a user and how to change the attributes of an already defined user. OpenEdition MVS information about the user is stored in a OMVS segment in the user profile. OpenEdition MVS information about the RACF group is stored in the OMVS segment in the RACF group profile. The following RACF TSO commands have been changed to allow adding or changing OpenEdition MVS information for a RACF user or RACF group profile:

- ADDUSER
- ALTUSER
- LISTUSER
- ADDGROUP
- ALTGROUP
- LISTGROUP

The ISPF RACF panels that are used to add, change and list RACF users and groups have been updated for OpenEdition MVS functions also.

6.4.1 Updates to RACF Panels for OpenEdition MVS

The following sequence of RACF panels is displayed after entering the requested selections on the various panels. The last panel in this list shows the various OMVS parameters that are specified for the selected user. In a standard ISPF/PDF environment the first panel will appear after entering **R** on the ISPF/PDF PRIMARY OPTION MENU.

Panel	Action
ICHP00	Select option 4 - user profiles
ICHP40	Select option 2 - change a user profile
ICHP42	Press Enter
ICHP42A	Enter yes for add or change optional
ICHP42A1	Enter the OMVS parameter with any character
ICHP42G	Shows the OMVS segment information you can add or change

The next list shows how to find OMVS group information:

Panel	Action
ICHP00	Select option 3 - group profiles
ICHP30	Select option 2 - and enter group name
ICHP32	Enter the OMVS parameter with any character
ICHP322	Shows the OMVS group information you can work with

6.4.2 User IDs (UIDs) and Group IDs (GIDs)

To authorize a RACF user to access OpenEdition MVS resources, you must add an OMVS user ID to the RACF user profile for an existing or new TSO/E user and connect each user to a RACF group that has a GID. If not already done so, you also have to add an OMVS group ID (GID) to a RACF group profile for an existing or new RACF group.

The UID and GID number value can be between 0 and 214783647.

Similar to the way some users with special RACF attributes have unrestricted ability to access resources and processes within MVS, there must be a so-called *superuser* defined for the OpenEdition MVS environment. Each user who has a UID = 0 is a superuser.

6.4.3 Effective UID and Effective GID

When a user is logged on to OpenEdition MVS, the UID from the user′s RACF user profile becomes the *effective UID* of his process. This effective UID is used to check the user authorization.

When a TSO/E user becomes an OpenEdition user, the GID from his current connect group becomes the *effective GID* of the user′s process. The user can access resources available to members of the user′s effective GID.

6.4.4 RACF List of Groups Checking and Supplemental Groups

When *RACF list-of-group checking* is active, a user can access an OpenEdition resource if it is available to members of any group the user is connected to and if the group has a GID in its RACF profile. The additional groups are called *supplemental groups*.

To activate the RACF list-of-group checking, specify the GRPLIST parameter on a SETROPTS command.

The maximum number of supplemental groups that can be associated with a process is 300.

6.4.5 Adding a New User with Access to OpenEdition MVS

To add a new user with access to the OpenEdition MVS environment, you should use the following commands. The user description is as follows:

```
Name      :    PETER
Group     :    PETER1
PROC      :    TSOPROC
PROGRAM   :    /bin/sh
HOME      :    /u/peter
```

To add the new user through a RACF TSO command, enter the following on the TSO command line:

```
ADDUSER PETER DFLTGRP(PETER1) NAME('PETER') PASSWORD(password)
 OMVS(UID(123) HOME('/u/peter') PROGRAM('/bin/sh'))
 TSO(ACCTNUM(acctnum) PROC(tsoproc) MAXSIZE(maximum_region_size))
```

If the (default) group (and the OMVS segment) does not already exist, you should use the following RACF command prior to entering the ADDUSER command:

```
ADDGROUP PETER1 SUPGROUP(supgroup) OMVS(GID(GID))
```

6.4.6 Giving an Existing User Access to OpenEdition MVS

You can alter the user profile for an already existing RACF user to add the information to the OMVS segment by using the following TSO RACF command:

```
ALTUSER PETER OMVS(UID(123) HOME('/u/peter') PROGRAM('/bin/sh'))
```

6.4.7 Listing the OMVS Segment Information for a User

To list the OMVS segment information in a user profile, use the following RACF TSO command:

```
LU PETER OMVS NORACF
```

The output of this LISTUSER command might look like:

```
USER=PETER

OMVS INFORMATION
----------------
UID= 123
HOME= /u/peter
PROGRAM= /bin/sh
```

The HOME information /u/peter is the home directory the user is connected to after he initialized the shell. The user can switch to another directory by using the OpenEdition shell command cd. PROGRAM tells you in which file the shell will be invoked.

RACF panels can also be used to list the OMVS segment in a user profile.

6.4.8 Giving an Existing Group Access to OpenEdition MVS

A user must belong to at least one group and can be connected to additional groups. When a user logs on to a TSO/E session, one of the groups is selected as the user's current group. For a user to be able to request OpenEdition MVS services and invoke the shell, the user's current RACF group must have an OpenEdition MVS group ID (GID) assigned to it.

For useful reports and auditing, assign a unique GID to each RACF group. To add a group ID (GID) to an already existing group, use the following RACF TSO command:

```
ALTGROUP PETER1 OMVS(GID(GID))
```

To list the OMVS segment information in a group profile, you can use the following command:

```
LG groupname OMVS NORACF
```

RACF ISPF panels can also be used to add and list the OMVS group information.

6.4.9 Home Directory

The security administrator specifies the pathname of the *home directory* for each user or process in the HOME field of the OMVS segment in the RACF user profile. When a user initializes the shell, the user's working directory is the home directory.

This home directory is the default for a OpenEdition cd command. The user can use the cd command to use another directory as the working directory.

On an open system, a working directory is normally defined in lowercase letters and usually has the user's TSO/E user ID as its name, for example '/u/peter'.

When the HOME directory is changed in a user profile, for example with the ALTUSER command, and this directory is not already defined, you will get the following message when the user tries to access open a OpenEdition shell:

```
FSUM2078I No session was started.  The home directory for this TSO/E user ID
does not exist or can not be accessed. +
FSUM2079I Function = sigprocmask, return value= FFFFFFFF, return code= 0000009C
reason code=0507014D
```

There are two different ways to define this home directory. It can be defined by either using a TSO/E command:

```
MKDIR 'directory_name' MODE(directory_permission_bits)
```

or through a OpenEdition shell command:

```
mkdir -m(permissions_for_directory) directory_name
```

The mode parameter ("MODE" in the TSO/E command and "-m") are used to specify the requested access authority for users in the OpenEdition MVS environment.

The permission bits are changed with the OpenEdition shell command chmod.

6.4.10 Field-Level Access for the OMVS Segment

To allow a user to see or change the OMVS field in a RACF user profile, an installation can set up field-level access. You can authorize a user to specified fields in any profile or to specified fields in the user's own profile. First you should define a profile for the OMVS segment fields UID, HOME and PROGRAM by using the following RACF TSO commands:

```
RDEFINE   FIELD   USER.OMVS.UID      UACC(NONE)
RDEFINE   FIELD   USER.OMVS.HOME     UACC(NONE)
RDEFINE   FIELD   USER.OMVS.PROGRAM  UACC(NONE)
```

In the next step you should permit users to access fields with the RACF PERMIT commands. &RACUID allows all users to access the fields in their own profiles with the specified authority. UPDATE access allows users to change the fields; READ access allows the users to read the fields.

```
PERMIT  USER.OMVS.UID     CLASS(FIELD) ID(&RACUID) ACCESS(READ)
PERMIT  USER.OMVS.HOME    CLASS(FIELD) ID(&RACUID) ACCESS(UPDATE)
PERMIT  USER.OMVS.PROGRAM CLASS(FIELD) ID(&RACUID) ACCESS(UPDATE)
```

Warning: Give only selected users UPDATE access to the UID field. A user with UPDATE access to the UID field can become a superuser by changing his UID to 0.

To allow authorization to the entire OMVS segment, for example for a OpenEdition MVS security administrator, the user would need UPDATE authority to the USER.OMVS.* profile in the FIELD class.

Similar access controls can be used to control access to the GID field in the OMVS segment of the group profile.

6.5 The Hierarchical File System (HFS)

A striking feature of OpenEdition MVS is the ability to make traditional MVS data sets accessible to POSIX applications and to make POSIX files accessible to traditional MVS applications. This is possible since POSIX data that is organized within a *hierarchical file system* is treated by MVS as a new kind of MVS data set: a *hierarchical file system data set*.

In POSIX, a file is a data object with certain attributes, such as access permissions. Files are collected together as a file system, with each file in a directory and directories arranged hierarchically up to the highest-level directory, which is called the *root directory*.

Figure 19 depicts the organization of the hierarchical file system.

Table 12 on page 78 shows how the environmentals of the hierarchical file system compares to MVS data sets.

6.5.1 Protecting HFS Data

The security rules for the files and directories in a HFS are stored within the file system itself in the *file security package* (FSP). HFS files and directories are protected by permission bit information which is kept within the FSP. Every file and directory has security information that specifies:

- Read, write, or search authority for a directory
- Read, write, or execute authority for a file
- Separate authorities for the owner of the file or directory (represented by a UID), the owning group (represented by a GID), and everyone else (like RACF′s universal access authority, or UACC)
- Audit options that the owner of the file or directory can control
- Audit options that the security auditor can control

Refer to 6.5.3, “File Access Permission Bits” on page 64 for a detailed description of these permission bits.

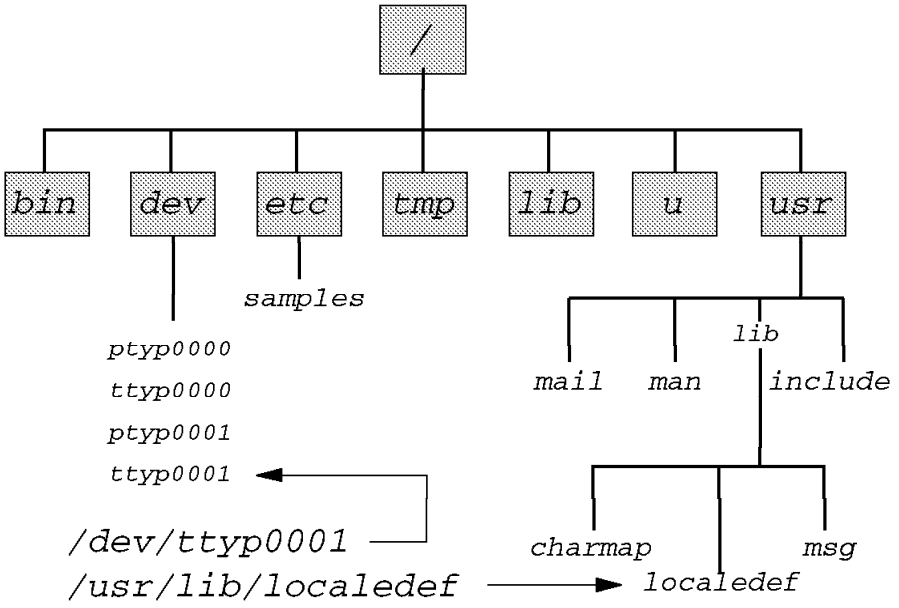

Figure 19. Organization of the OpenEdition File System

6.5.2 RACF Callable Services

When a security decision is needed, the file system calls RACF and supplies the FSP. RACF makes the decision, does any auditing, and returns control to the file system. The following should be taken into account:

- RACF does not provide commands to maintain the FSP.
- RACF provides SAF services that does FSP maintenance. OpenEdition provides commands that invoke these SAF services.
- OpenEdition shell commands should be used to maintain the FSP.

Examples of SAF services, or so called *callable services*, are:

makeFSP This service builds an FSP in the area provide by the caller.

ck_access This service determines whether the current process has the requested access to the element (directory or file) of a pathname whose FSP is passed.

R_chmod This service checks whether the calling process is authorized to change the mode of the specified file that is identified by the input FSP and, if so, changes the permission bits.

R_chaudit This service verifies that the user has authority to change the audit options for the specified file and, if so, sets the audit bits from the input audit options parameter.

Normal customer applications using services or functions of OpenEdition cannot call the RACF callable services directly. They must use the OpenEdition MVS callable services instead.

For a detailed description of the RACF callable services and for a complete mapping of the file security package, see *RACF Callable Services*.

6.5.3 File Access Permission Bits

File access permission bits that accompany each file (or directory) provide discretionary access control. These bits determine the type of access a user has to a file or directory.

The access permission bits are set for three classes of users:

- Owner class: Any process with an effective UID that matches the UID of the file.
- Group class: Any process with an effective GID or supplemental GID that matches the GID of the file.
- Other class: Any process that is not in the owner or group class.

When a user's process accesses a file, the system determines the class of the process and then uses the permission bits for that class to determine if the process can access the file. For a file, a process can be in only one class. The class for a process can be different for each file or directory.

The permission bits are split into three groups of three characters. The first group of three characters describes the owner permissions; the second describes group permissions; the third describes other permissions. Characters that can be used on fixed positions in each of the groups of three characters, are:

Table 10. File Access Types and Permission Bits

Pos.	Char.	Access Type	Permission for File	Permission for Directory
1	r	Read	Permission to read or print the contents	Permission to read, but not search, the contents
2	w	Write	Permission to change, add to, or delete from the contents	Permission to change, add, or delete directory entries
3	x	Execute or Search	Permission to run the file. This permission is used for executable files.	Permission to search the directory
any	-	no access		

Permission bits are usually presented in a 3-digit octal number, where the first digit describes owner permissions, the second digit describes group permissions, and the third digit describes permissions for all others.

For each type of access, owner, group, and other, there is a corresponding octal number:

0 No access (---)

1 Execute-only access (--x)

2 Write-only access (-w-)

3 Write and execute (-wx)

4 Read-only access (r--)

5 Read and execute access (r-x)

6 Read and write access (rw-)

7 Read, write, and execute access (rwx)

Some typical 3-digit permissions are specified in octal in this way:

666 Owner (rw-) group(rw-) other(rw-)

700 Owner (rwx) group(---) other(---)

755 Owner (rwx) group(r-x) other(r-x)

777 Owner (rwx) group(rwx) other(rwx)

6.5.4 Default Permissions Set by the System

The system assigns default access permissions for files and directories at creation time. The setting is depending on type of command or facility that is used and on the type of file or directory that is created.

The following table shows some default permissions set by the system:

Table 11. Default Permissions Set by the System

Task	Using	Default Permissions
Create directory	shell cmd **mkdir**	In octal form: 777
Create directory	TSO/E MKDIR	In octal form: 755
Create file	OEDIT command	In octal form: 700
Create file	**ed** editor	In octal form: 666
Create file	**cp** command	Set the output file permission to the input file permissions.

For more information on the default permissions settings, refer to: *MVS/ESA OpenEdition MVS User's Guide*.

6.5.5 Temporary Access

An executable file, which is a file containing a shell script or a program, can have an additional attribute in the permissions. This permission setting is used to allow a program temporary access to the files that are not normally accessible to other users.

An **s** or **S** can appear in the *execute* permission position; this permission bit sets the *effective user ID* or *effective group ID* of the user process executing a program to that of the file whenever the file is run.

s In the owner permissions section, this indicates that both the *set_user_ID* (S_ISUID) bit is set and execute (search) permission is set.

In the group permissions section, this indicates that both the *set_group_ID* (S_ISGID) bit is set and execute (search) permission is set.

S In the owner permissions section, this indicates the *set_user_ID* (S_ISUID) bit is set, but the execute (search) bit is not.

In the group permissions section, this indicates the *set_group_ID* (S_ISGID) bit is set, but the execute (search) bit is not.

A good example of this behavior is the `mailx` utility. This utility is used to send and receive electronic mail messages to and from users that operate on the same system.

A user sending mail to another user on the same system is actually appending the mail to the recipient's mail file, even though the sender does not have the appropriate permissions to do this; the mail utility `mailx` does have the appropriate permissions.

6.5.6 Changing Permissions for Files and Directories

The shell command `chmod` can be used to set or change permissions for files and directories. To change permissions, you must be the owner of the file or directory. The `chmod` can be specified as follows:

`chmod` *mode pathname*

The `chmod` mode can be specified in two different forms: in a symbolic form or as a octal value.

The symbolic form of the *mode* argument has the form:

who op permission

The *who* value is any combination of the following:

u Sets owner (user or individual) permissions

g Sets group permissions

o Sets other permissions

a Sets all permissions; this is the default. If a *who* is not specified, the default is **a** and the file creation mask is applied.

The *op* of a symbolic mode is an operator that tells `chmod` to turn the permissions on or off. The possible values are:

+ Turns on a permission

- Turns off a permission

= Turns on the specified permissions and turns off all others

The *permission* part of a symbolic mode is any combination of the following:

r Read permission

w Write permission

x Execute a file or search permission for a directory

The following indicators have a special meaning and might not be valid on all positions.

s If in the owner permissions section, the *set_user_ID* bit is on; if in the group permissions section, the *set_group_ID* bit is on.

t This represents the *sticky_bit*. This bit causes OpenEdition MVS to search for the program in the user's STEPLIB, the link pack area, or in the link list concatenation.

Typically, octal permissions are specified with three or four numbers in the positions 1234. Each position indicates a different type of access:

Position 1 Sets permission for the *set_user_ID* on access, *set_group_ID* on access, or the *sticky_bit*.

Position 2 Sets permissions for the owner of the file.

Position 3 Sets permissions for the group that the owner belongs to.

Position 4 Sets permissions for others.

For position 1 you can specify:

0 Off.

1 Sticky bit is on. Only a superuser can specify this.

2 Set_group_ID on execution.

3 Set_group_ID on and set the sticky bit on. Only a superuser can specify this.

4 Set_user_ID on execution.

5 Set_user_ID on and set the sticky bit on. Only a superuser can specify this.

6 Set_user_ID and set_group_ID on execution.

7 Set_user_ID and set_group_ID on execution and set the sticky bit. Only a superuser can specify this.

For positions 2, 3, and 4 you can specify:

0 No access (---)

1 Execute-only access (--x)

2 Write-only access (-w-)

3 Write and execute (-wx)

4 Read-only access (r--)

5 Read and execute access (r-x)

6 Read and write access (rw-)

7 Read, write, and execute access (rwx)

To specify permissions for a file or directory, you use at least a 3-digit octal number, where the first digit describes owner permissions, the second digit describes group permissions, and the third digit describes permissions for all others.

For example, both commands set the same permissions for file2:

```
chmod a=rwx  file2
chmod 0777  file2
```

This command turns on read, write, and execute permissions for the owner, the group the owner belongs to, and others, and turns off the *set_user_ID* bit, *set_group_ID* bit, and the *sticky_bit*.

6.5.7 Setting the File Mode Creation Mask

When a file is created, it is assigned initial access permissions by the system. If a user wants to control the permissions that a program can set when it creates a file or directory, he can set a *file mode creation mask*, using the umask shell command.

A user can set this file mode creation mask for one shell session by entering the umask command interactively, or he can make the umask command part of his login.

When the user is setting the mask, he is setting limits on allowable permissions: he is implicitly specifying which permissions are not to be set, even though the calling program may allow those permissions. When a file or directory is created, the permissions set by the program are adjusted by the unmask value. The final permissions set are the program's permissions minus what the umask values restrict. To use the umask command for a single session, enter:

umask *mode*

and specify the mode in either of the formats used by the chmod: symbolic (rwx) or octal values.

The symbolic form expresses what can be set and what is allowed, while octal values express what cannot be set and what is disallowed. For example, both of the following commands set the same umask:

```
umask a=rx
umask 222
```

The modes are explained as follows:

a=rx	**Explanation (symbolic form expresses what is allowed**)
a	Set all permissions, this include the owner, group, and others
=	Turn on the specific permissions and turn off all others
rx	read and execute permissions

222	**Explanation (octal values express what is disallowed**)
2	Write-only access for the owner
2	Write-only access for the group
2	Write-only access for others

Both modes indicate that the owner, group, and others have read and execute permission.

6.5.8 Display the Permission Bits

The OpenEdition shell commands ls -l and ls -W enable the user to display the permission bits for a specified file or directory.

Note: Be aware of using capital and small letters in OpenEdition shell commands since they are case-sensitive; capital letters do not always mean the same thing as small letters.

Here is a sample output from a ls -l command along with an explanation.

```
drwxr-x--x     3  OMVSKERN  SYS1     0 Jun 6  14:49 tmp
```

The permission bits appear as 10 characters (drwxr-x--x).

d Identifies the type of a file or directory, in this case a directory is displayed.

rwx Any process with a UID that matches the UID of the directory has read, write and search authority for this directory.

r-x Any process with a GID that matches the GID of the directory has read, and search authority for this directory.

--x Any other process has search authority for this directory.

After the permissions are set, `ls` displays the following about this directory:

3 The number of links to the file

OMVSKERN The name of the owner of the directory (or file)

SYS1 The name of the group that own the directory (or file)

0 The size of the file, expressed in bytes (in case a file is displayed)

Jun 6 14:49 The date when the directory was created (or for a file, the date the file was last changed)

tmp The name of the directory (or file)

If the specified *pathname* is a directory, like in the previous example, `ls` displays information on every file in that directory (one file per line). A sample output might look like:

```
drwxr-x--x     3  OMVSKERN  SYS1     0 Jun 6  14:49 tmp
drwxrwxrwx     2  PETER     PETER1   4 Jun 7  09:12 usr
-rwxr--r--     1  PETER     PETER1 124 Jun 7  10:24 bin
-rwxrw----     1  PETER     PETER1 572 Jun 7  16:32 abc
```

6.5.9 Access Violations

If the user does not have sufficient access for the requested file or directory, a normal RACF message ICH408I is displayed on the system console. This message is issued when RACF detects an unauthorized request (a violation) made by a user or a process. The user and the group indicated in the first line of the ICH408I message are the execution user ID and group ID under which the process did execute.

The line “INSUFFICIENT AUTHORITY TO” shows the OpenEdition callable service name that was causing the error. In the sample below, it is the *open* callable services. The “open” callable service gains access to a file. Also the file name, the requested access type, in this case read and write access, and the access class that is used to check the users authority, is displayed. In this example the class “OTHER” is used since the user’s UID, as well as the user connected group GID did not match the UID or GID that was set for this file.

```
ICH408I USER(PETER) GROUP(PETER1) NAME(HILGER)
        /dev/ttyp0000 CL(FSOBJ) FID(01C1D3D3F0F0F1.....)
        INSUFFICIENT AUTHORITY TO OPEN
        ACCESS INTENT(RW-) ACCESS ALLOWED(OTHER ---)
```

Note: The Function ID (FID) value is not important for the user. This field contains debug information that is used by IBM service and support centers.

6.6 Auditing an OpenEdition Environment

The task of an auditor basically consists of verifying that the principles set forth in an installation's security policy are not compromised. In an OpenEdition environment which uses RACF as its access control program, there are two main tasks to perform:

- Verifying that the RACF profiles have the proper contents (OMVS segments in the user and group profile and logging options in particular)
- Use the security logs to follow up on detected violations and to detect abnormal behavior by authorized users

The audit information can be quite extensive and is found in the RACF database, the RACF security log, and in the logs produced by applications that use RACF services.

The problem facing an auditor is mostly that of being able to reduce the amount of information to something that can be easily analyzed, and perhaps more important to be able to find the needles in some very large haystacks.

The tools available to do auditing are the normal RACF commands, the RACF Report Writer command and applications, such as the Service Level Reporter (SLR) and the SystemView Enterprise Performance Data Manager (EPDM).

With RACF Version 1 Release 9 also came the *RACF database unload utility* which gave RACF administrators and auditors a entire new source of information. By loading the sequential output file from the utility into a relational database, such as IBM DATABASE 2 (DB2), they now could perform adhoc queries on the RACF database, without the risk of impairing system performance.

For the auditor to analyze RACF security logs and the SMF data in particular, the *RACF report writer* command (RACFRW) has traditionally been the main vehicle. However, auditors have long been complaining about the readability of the RACFRW output, about the inability to select events only after they exceed a given number, and about the fact that the RACFRW does not limit a group auditor to the events produced within the scope of the auditor. Most installations have, therefore, written their own post processor programs to do additional processing based on the RACFRW output.

The RACF report writer does not completely support reporting of the OpenEdition audit data. It will process the SMF records but it won't report them completely, and it won't allow specific selection of the records. The report writer is no longer the IBM-recommended utility for processing RACF audit records.

With the availability of RACF Version 2 Release 1 also came a change in the auditing functions for the system. The RACFRW command has been functionally stabilized on the RACF Version 1 Release 9.2 level, and all the new event codes can only be handled using the *RACF SMF data unload utility*.

This utility converts RACF SMF records into a sequential dataset (flat file). This dataset can be sorted and records selected based on various selection criteria. The unloaded SMF records can also be loaded into a relational database and be processed with suitable query languages. Figure 20 depicts the new RACF auditing environment.

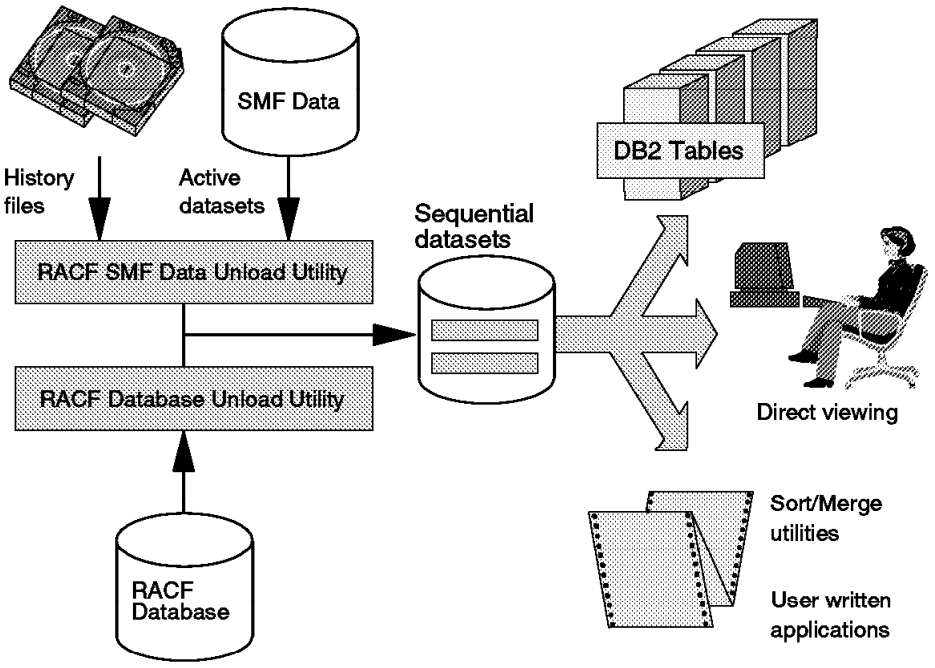

Figure 20. New RACF Auditing Environment

There is a slight problem connected with the use of relational databases: users have to be taught the Structured Query Language (SQL), the Query Management Facility (QMF) or some other query language if they want to be able to perform their own adhoc queries. The alternative is for someone to build an application with a set of predefined reports which can easily be adapted to fit the individual installation.

For more information on auditing, refer to *Enhanced Auditing Using the RACF SMF Data Unload Utility*. This document describes a number of RACF auditing tools that are based on the RACF SMF data unload utility and on the RACF database unload utility. The primary audience is RACF security auditors.

The Interactive System Productivity Facility (ISPF) program product is chosen to drive this auditors application, and to perform the actual queries, a combination of REXX EXECs and QMF queries is used.

The *auditing application* that is described in *Enhanced Auditing Using the RACF SMF Data Unload Utility* consists of the following parts:

- A auditing application that uses the ISPF, REXX EXECs, and QMF on MVS. This application is based on the RACF SMF data unload utility.
- A auditing application that runs in a OS/2 environment using DB2/2, DDCS/2, and the Visualizer Query for OS/2. This application is also based on the RACF SMF data unload utility.
- The enhanced reporting application that uses ISPF, REXX EXECs, and QMF on MVS. This application is based on the RACF database unload utility.

A copy of the source for these ISPF applications, the OS/2 based application, and REXX procedures described in *Enhanced Auditing Using the RACF SMF Data Unload Utility* will be administrated by the International Technical Support Organization in Poughkeepsie. In this case, “administration” means the tools

package will be refreshed when required, but the code remains on a “best support” basis. The package is available on the MVSTOOLS disk under the name ASKO-2.

If you are interested in these applications, please contact your local IBM representative.

6.6.1 Auditing OpenEdition MVS Events

The security auditor uses reports formatted from RACF *system management facilities* (SMF) records to check successful and failing accesses to OpenEdition MVS resources. An RACF SMF record can be written at each point where RACF is asked to make a security decision.

RACF Version 2 Release 1 provides six new classes that are used to control auditing of the OpenEdition security events. These classes have no profiles and they do not have to be activated to control auditing. You should use the SETROPTS command to specify the auditing options for the classes.

The following new classes are defined:

- DIRSRCH
- DIRACC
- FSOBJ
- FSSEC
- PROCAT
- PROCESS

Note: These classes are for audit purposes only.

DIRSRCH Controls auditing of directory searches.

DIRACC Controls auditing for access checks for read/write access to directories.

FSOBJ Controls auditing for all access checks for files and directories.

FSSEC Controls auditing for changes to the security data (FSP) for file system objects.

PROCESS Controls auditing of changes to the UIDs and GIDs of processes.

PROCAT Controls auditing of functions that look at data from other processes or affect other processes.

Auditing can be controlled by using the commands SETROPTS LOGOPTIONS and SETROPTS AUDIT.

LOGOPTIONs(*auditing_level*(*class_name*)) audits access attempts to the resources in the specified class according to the auditing-level specified. Auditing-level specifies the access attempts to be logged for the class_name. These options are processed in the order listed below.

ALWAYS All access attempts to resources protected by the class are audited.

NEVER No access attempts to resources protected by the class are audited. (All auditing is suppressed.)

SUCCESSES All successful access attempts to resources protected by the class are audited.

FAILURES All failed access attempts to resources protected by the class are audited.

DEFAULT Auditing is controlled by the profile protecting the resource, if a profile exists.

For example, an installation can specify:

```
SETROPTS LOGOPTIONS(ALWAYS(DIRSRCH,DIRACC))
```

AUDIT(*class_name*) specifies the names of the classes to which you want RACF to perform auditing. For the classes you specify, RACF logs all uses of the RACROUTE REQUEST=DEFINE SVC and all changes made to profiles by RACF commands. You can use the AUDIT option to control auditing for the FSOBJ and the PROCESS classes.

For example, an installation can specify:

```
SETROPTS AUDIT(FSOBJ,PROCESS)
```

The following are events for which audit records are always written:

- When a user not defined as an OpenEdition user tries to dub a process
- When a user who is not a superuser tries to mount or unmount the file system
- When a user tries to change a home directory
- When a user tries to remove a file, hard link, or directory
- When a user tries to rename a file, hard link, symlink, or directory
- When a user tries to create a hard link

There is no option to turn off these audit records.

Notes:

1. Actions by superusers can be audited, except when the superuser also has RACF privileged authority.
2. Actions by RACF privileged authority users are not audited.

6.6.2 Audit Options for File and Directory Levels

An installation can also specify auditing at the file level in the file system. This can be activated by:

- Specifying: SETROPTS LOGOPTIONS(DEFAULT(DIRSRCH,DIRACC,FSOBJ))
- Using the OpenEdition shell command `chaudit` to specify the audit options for individual files and directories

The following audit options for file and directory levels are stored inside the HFS along with the *file permission bits*:

- Don't_audit
- Audit_access_allowed
- Audit_access_failed
- Audit_all_access

The command can be used to specify either user audit options or auditor audit options. To specify user audit options, you must be a superuser or the owner of the file. To specify auditor options, you must have RACF AUDITOR authority. If both user and auditor options are set, RACF merges the options and audits all the set options. The format of the shell command is:

`chaudit` *`options attr pathname`*

In *options*, you can specify the type of resource, for example:

-F Audit characteristics of all files in the directory that is specified in the pathname are changed. Sub-directory audit characteristics are not changed.

-d Audit characteristics of all sub-directories in the directory that is specified in the pathname are changed. File audit characteristics are not changed.

-a Auditor-requested audit attributes are to be changed for the files or directories that are specified in the pathname. If **-a** is not specified, user-requested audit attributes are changed.

-i Does not issue error messages concerning file access authority, even if `chaudit` encounters such errors.

The *attr* field has the following format:

`operation op auditcondition`

- The *operation* value is any combination of the following:

  r Audit read attempts

  w Audit write attempts

  x Audit execute attempts

- You can also specify an *op* part that will act as an operator. The possible values are:

  + Turns on specified audit conditions

  - Turns off specified audit conditions

  = Turns on the specified audit conditions and turns off all others

- The *audit condition* is any combination of the following:

  s Audit on successful access if the audit attribute is on

  f Audit on failed access if the audit attribute is on

An installation can specify multiple symbolic *attr* values if they separate them with commas.

To change the audit attributes for **file1** so that all successful and unsuccessful file accesses are audited, you should enter:

```
chaudit rwx=sf file1
```

To change the audit attributes for **file3** so that unsuccessful file read accesses are not audited but successful write accesses are audited, you should enter:

```
chaudit r-f,w+s file3
```

6.7 Administrative Tasks Using the OpenEdition MVS ISPF Shell

Although the user can invoke administrative commands for the OpenEdition environment from a TSO/E command line, most users invoke TSO/E commands or programs from an ISPF/PDF menu.

The OpenEdition MVS ISPF shell is a panel interface that can be used instead of the TSO/E commands or shell commands to perform certain tasks.

For example, ISPF Edit provides a full-screen editor a user can use to create and edit HFS files. A user can access *ISPF file edit* from the ISPF menu (if a menu option is installed), or by typing the TSO/E OEDIT command at the TSO/E READY prompt or from the shell command line.

In order to make certain TSO/E commands (such as OEDIT, OBROWSE, and ISHELL) and some shipped REXX EXECs available to users, the installation must concatenate a number of target libraries to the appropriate ISPF data definitions (ddnames). The TSO/E commands OEDIT, OBROWSE, and ISHELL should be added to the ISPF/PDF PRIMARY OPTION MENU. For a detailed description of this implementation, please refer to *MVS/ESA Planning: OpenEdition MVS*.

The TSO/E command ISHELL invokes the OpenEdition MVS ISPF shell. The ISPF shell can also be invoked from a ISPF menu if this option is specified within ISPF. Figure 21 depicts the basic *OpenEdition MVS ISPF shell* panel that will be displayed after the ISHELL is invoked.

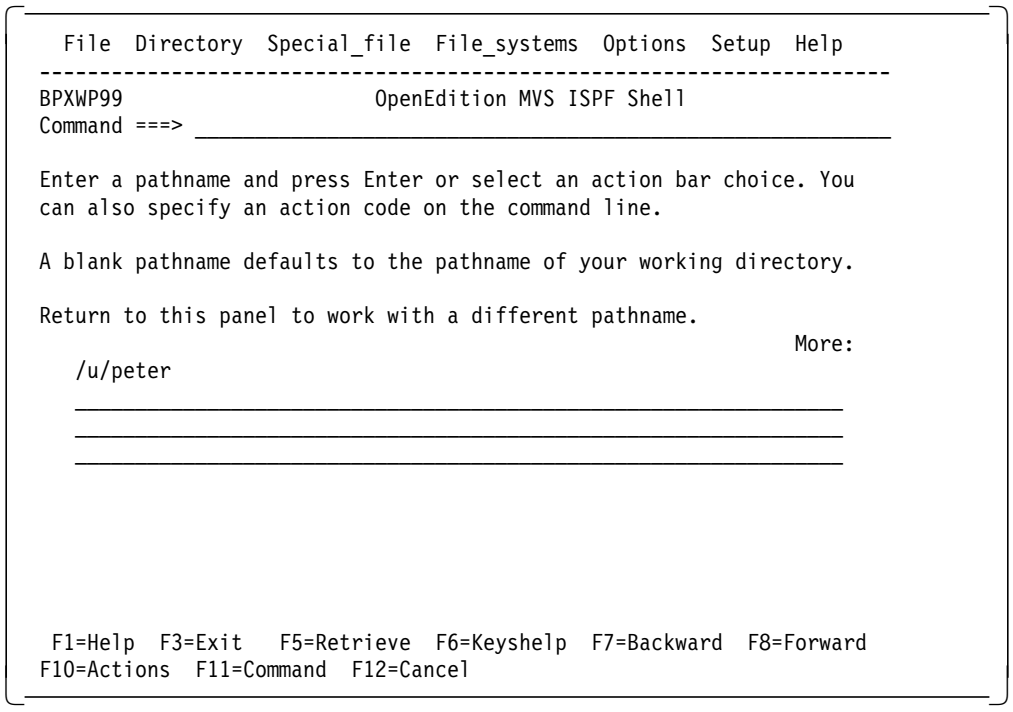

Figure 21. OpenEdition MVS ISPF Shell

The action bar at the top of this panel gives several options to select:

- File
- Directory
- Special file

- File systems
- Options
- Setup
- Help

Work in the OpenEdition ISPF shell is a two step sequence:

1. Select an *object* -the pathname of a new or existing file
2. Select an action for that object

In the center of the panel are three lines that can be used to specify the pathname of a file that needs to be selected. It can be a name of an existing file or a new file that is created in this step.

Another way to select a pathname or file is by selecting **Directory** on the action bar. This will display a pulldown panel that enables you to select a directory. All the files in a directory are displayed in a list; you can then select the file.

In the lower part of the panel is a command line that can be used to specify an *action_code*, a one-character code that specifies an action that you want to perform on the pathname that is specified.

The action bar on the main panel can also be used to select a specific action for a specified object. To use the action bar, position the cursor under one of the choices and press Enter. A pulldown panel with a list of actions is displayed. Figure 22 depicts the **Directory** pulldown.

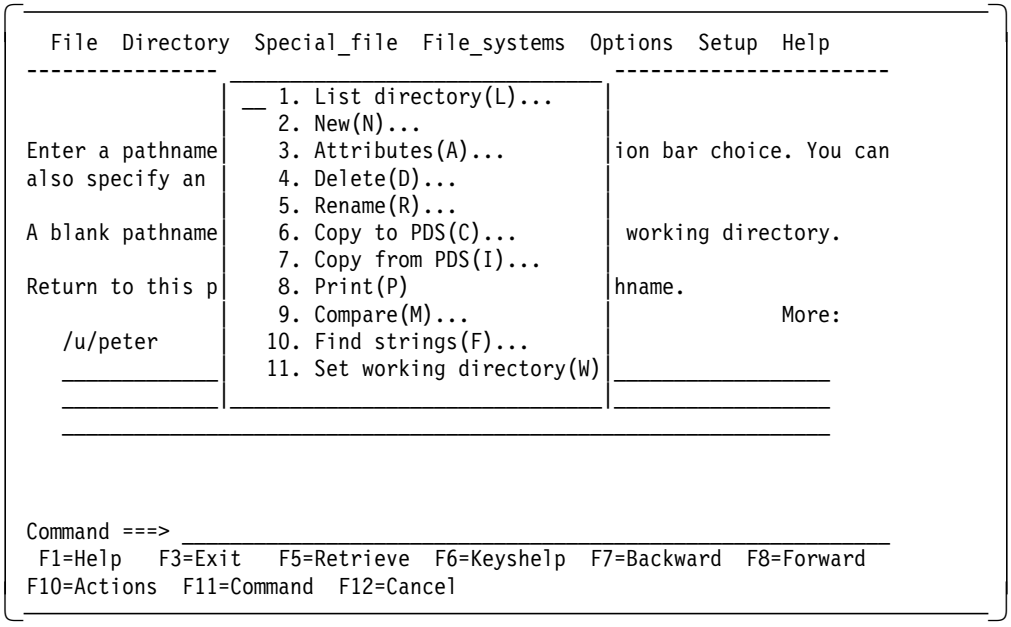

Figure 22. A Pulldown Panel Selected from the Action Bar

6.8 Parallels between the TSO/E Environment and the Shell Environment

Figure 23 depicts the facilities that can be used to do data, file, and security management in a OpenEdition MVS environment. End users and administrators can use TSO/E commands, ISPF panels, and OpenEdition shell commands to perform their tasks. Users can pick the facility they are most familiar with since most of the needed tasks can be performed with every facility. An interactive user can switch back and forth between the facilities.

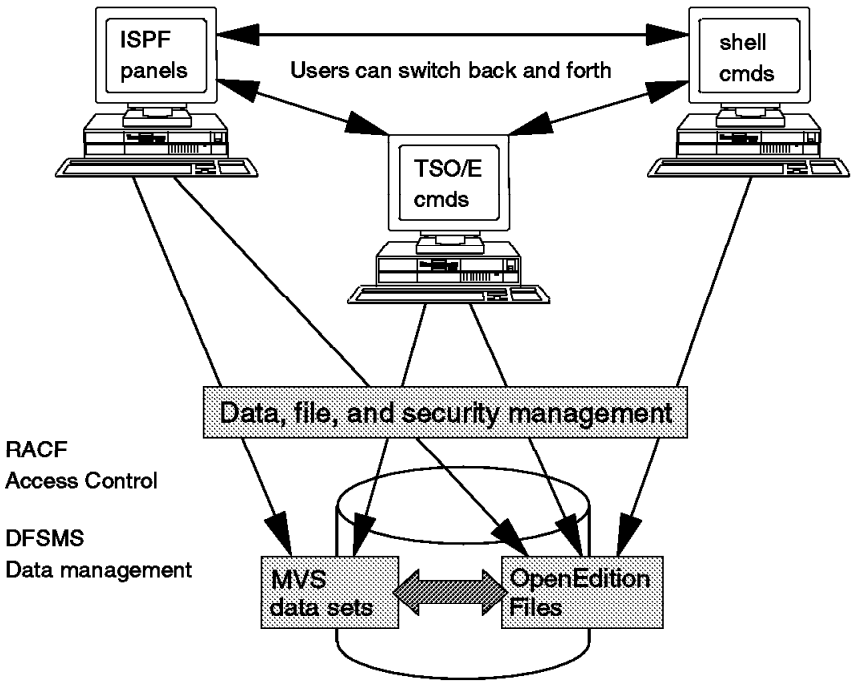

Figure 23. Data, File, and Security Management

Table 12 on page 78 indicates how basic end-user tasks can be performed in the MVS TSO/E environment and in the OpenEdition shell environment.

This table compares end-user activities to allocate and use data sets in the TSO/E environment and end-user activities to allocate and use files in a OpenEdition environment. It also shows some of the activities that have to be preformed by a security administrator, or at least someone with some extra authority that allows him to do some security work.

Table 12 (Page 1 of 2). Comparing POSIX Files and MVS Data Sets

POSIX definition	**MVS equivalent**
POSIX definitions are made by using: • OpenEdition shell commands • TSO/E commands • OpenEdition ISPF panels	MVS data set management is done by: • TSO/E commands • ISPF/PDF panels • IDCAMS commands
Catalog management	
Root directory and sub-directories SYS1.PARMLIB(BPXPRMxx) ROOT FILESYSTEM(*root directory*) TYPE(HFS) MODE(RDWR) TSO/E MKDIR '*directory_name*' MODE Or the shell command: **mkdir -m** *permissions directory_name*	Master catalog and alias SYS1.PARMLIB(LOADxx) SYSCAT AMS command DEFINE USERCATALOG - (NAME(*catalog_name*) - VOLUME(*volser*) - ICFCATALOG STRNO(3)) AMS command DEFINE ALIAS - (NAME(*high-level-qualifier*) RELATE (*catalog_name*)) - CATALOG (*catalog_name*)
File or data set allocation	
TSO/E ALLOCATE DS(*filename*) DSNTYPE(HFS) SPACE(*space*) TSO/E MOUNT FILESYSTEM (*filename*) TYPE (HFS) MOUNTPONT ((*sub*)-*directory*) OpenEdition files are byte-oriented (defined by the application)	TSO/E ALLOCATE DS(*filename*) DSORG(*dsorg PS or PO*) RECFM(*record format*) LRECL(*record length*) BLKSIZE(*block size*) CATALOG MVS data sets are record-oriented
Access control	
• File access permission bits • Default permissions set by the system • User control of permissions through *file mode creation mask* shell command: umask *mode* • Shell command to change the mode of a file or directory: chmod *mode pathname*	• Global access table • Access lists in resource profiles: RACF ADDSD *profile_name* OWNER(*owner user or group ID*) UACC(*universal access authority*) AUDIT(*access_attemps* (*audit_access_level*)) RACF PERMIT *profile_name* CLASS(DATASET) ID(*user ID*) ACCESS(*access_type*)

Table 12 (Page 2 of 2). Comparing POSIX Files and MVS Data Sets

POSIX definition	MVS equivalent
Access control levels	
Access **Permission for file or directory** **READ** read or print file contents read or print directory contents, but not search **WRITE** change, add to, or delete from the file contents change, add, or delete directory entries **EXECUTE or SEARCH** run the file (used for executable files) search the directory	**Access** **UACC or in data set profiles** **NONE** no access allowed **READ** read access only **UPDATE** read and write access to datasets does not authorize users to delete, rename, move, or scratch the data set **CONTROL** retrieve, update, insert, or delete records from the VSAM data set. **ALTER** read, update, delete, rename, move, or scratch the data set **EXECUTE** load and execute from load library, but not read or copy, programs (load modules) in the library Anyone who has READ, UPDATE, CONTROL, or ALTER authority to a protected data set can create a copy of it.
List the autorizations	
Shell command to list file and directory names and attributes: **ls** options *pathname* **ls -l** Displays the permissions; if the specified pathname is a directory, it displays information on every file in that directory. **ls -W** Enables the audit bits to be displayed. It turns on the **-l** option.	RACF LISTDSD DATASET(*profile_name*) AUTHUSER DSNS
Auditing	
SETROPTS LOGOPTIONS SETROPTS AUDIT User audit option and auditor audit options: chaudit *options attr pathname*	SETROPTS LOGOPTIONS SETROPTS AUDIT AUDIT(*access_attempt(audit_level*)) on ADDSD or ALTDSD GLOBALAUDIT(*access_attempt(audit_level*)) on ALTDSD
File editing	
• Shell command **ed** *file_name* • ISPF - OpenEdition Edit • TSO/E - OEDIT *pathname*	• TSO/E EDIT *data_set_name* • ISPF/PDF option EDIT

Appendix A. Started Procedure Tables Report

This appendix provides information on the RACF *data security monitor* (DSMON) and some details on a specific report from DSMON, the *started procedure tables* report.

A.1 DSMON - Sample JCL

The RACF data security monitor (DSMON) produces a report of the currently active *started procedures table*.

If the *dynamic started procedures table support* is active, DSMON creates a two-part output listing, the first part contains the entries from the active STARTED class, the second part contains the entries from the active ICHRIN03 module.

The following JCL can be used to create the started procedures report:

```
//STEP1    EXEC PGM=ICHDSM00
//SYSPRINT DD SYSOUT=*
//SYSUT2   DD SYSOUT=*
//SYSIN    DD *
  FUNCTION RACSPT
/*
```

Notes:

1. The use of the RACF data security monitor requires the AUDITOR attribute or explicit authorization for ICHDSM00 in the PROGRAM class.
2. The DDNAME SYSUT2 contains the DSMON output data. If you want to route the output into a data set, use the following data set characteristics:
   - DSORG=PS
   - RECFM=FBA
   - LRECL=133

When the STARTED class is *active*, the report consists of two parts:

1. The listing from the active STARTED class
2. The listing from the active ICHRIN03 module

When the STARTED class is *not active*, the report just consists of the ICHRIN03 part.

See A.2, “DSMON Output” on page 82 for a sample of the DSMON output report.

A.2 DSMON Output

This topic provides an example of a RACF started procedures table report.

```
RACF DATA SECURITY MONITOR
  R A C F   S T A R T E D   P R O C E D U R E S   T A B L E   R E P O R T
FROM PROFILES IN THE STARTED CLASS:
--------------------------------------------------------------------------------
PROFILE               ASSOCIATED  ASSOCIATED
NAME                  USER        GROUP       PRIVILEGED  TRUSTED  TRACE
--------------------------------------------------------------------------------
AOFAPPL.* (G)         STC         SYS1        YES         NO       NO
AOFASSI.* (G)         STC         SYS1        YES         NO       NO
CNMAPROC.* (G)        STC         SYS1        YES         NO       NO
CNMASSI.* (G)         STC         SYS1        YES         NO       NO
DFHSM.* (G)           STC         SYS1        YES         NO       NO
DVGSTQH.* (G)         STC         SYS1        YES         NO       NO
IRRDPTAB.* (G)        STC         SYS1        YES         NO       NO
ISPFLMF.* (G)         STC         SYS1        NO          NO       NO
JES2.* (G)            STC         SYS1        YES         NO       NO
LLA.* (G)             STC         SYS1        YES         NO       NO
LOGREC.* (G)          STC         SYS1        YES         NO       NO
NET.* (G)             STC         SYS1        YES         NO       NO
NETO.* (G)            STC         SYS1        YES         NO       NO
PROCXYZ.* (G)         STC         SYS1        NO          NO       NO
RACF.* (G)            STC         SYS1        NO          NO       NO
RMF.* (G)             STC         SYS1        YES         NO       NO
SAMON.* (G)           STC         SYS1        YES         NO       NO
SMF.* (G)             STC         SYS1        YES         NO       NO
TSO.* (G)             STC         SYS1        YES         NO       NO
UCBMOUNT.* (G)        STC         SYS1        YES         NO       NO
** (G)                STC         SYS1        NO          NO       YES

RACF DATA SECURITY MONITOR
  R A C F   S T A R T E D   P R O C E D U R E S   T A B L E   R E P O R T
FROM THE STARTED PROCEDURES TABLE (ICHRIN03):
--------------------------------------------------------------------------------
PROCEDURE       ASSOCIATED      ASSOCIATED
NAME            USER            GROUP           PRIVILEGED      TRUSTED
--------------------------------------------------------------------------------
PROCXYZ         STC             SYS1              NO              NO
JES2            STC             SYS1              YES             NO
NET             STC             SYS1              YES             NO
NETO            STC             SYS1              YES             NO
TSO             STC             SYS1              YES             NO
IRRDPTAB        STC             SYS1              YES             NO
DFHSM           STC             SYS1              YES             NO
RMF             STC             SYS1              YES             NO
LLA             STC             SYS1              YES             NO
SMF             STC             SYS1              YES             NO
LOGREC          STC             SYS1              YES             NO
AOFAPPL         STC             SYS1              YES             NO
CNMAPROC        STC             SYS1              YES             NO
AOFASSI         STC             SYS1              YES             NO
CNMASSI         STC             SYS1              YES             NO
ISPFLMF         STC             SYS1              NO              NO
UCBMOUNT        STC             SYS1              YES             NO
SAMON           STC             SYS1              YES             NO
DVGSTQH         STC             SYS1              YES             NO
RACF            STC             SYS1              NO              NO
*               STC             SYS1              NO              NO
```

Appendix B. Convert ICHRIN03 to STARTED Class Profiles

This appendix provides information on how to convert the *started procedures table* (ICHRIN03) into profiles in the STARTED class.

RACF 2.1 supplies a REXX EXEC to convert entries from the started procedures table (ICHRIN03) into resource definition commands for the RACF class STARTED. The EXEC resides in member ICHSPTCV in data set SYS1.SAMPLIB.

The following JCL can be used to run the EXEC:

```
//DSMON    EXEC PGM=ICHDSM00
//STEPLIB  DD DISP=SHR,DSN=SYS1.LINKLIB,
//            UNIT=SYSALLDA,VOL=SER=xxxxxx
//SYSPRINT DD SYSOUT=*
//SYSUT2   DD DISP=(,PASS),DSN=&&TEMP,
//            UNIT=VIO,RECFM=FB,LRECL=133
//SYSIN    DD *
  FUNCTION RACSPT
/*
//SPTCV    EXEC PGM=IKJEFT01,PARM='%ICHSPTCV'
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD DUMMY
//SYSEXEC  DD DISP=SHR,DSN=SYS1.SAMPLIB,
//            UNIT=SYSALLDA,VOL=SER=xxxxxx
//INFILE   DD DSN=&&TEMP,DISP=(OLD,DELETE)
//OUTFILE  DD DISP=(NEW,CATLG),DSN=userid.STARTED.DEFS,
//            UNIT=SYSALLDA,VOL=SER=yyyyyy,
//            SPACE=(TRK,(1,1)),
//            RECFM=VB,LRECL=255,BLKSIZE=6400
//*
```

Step Name	Description
DSMON	Calls the RACF data security monitor (DSMON).
	Input: The started procedures table (ICHRIN03).
	Output: The started procedures table report. The report is placed into a temporary data set and is then passed to the next step for further processing.
SPTCV	Calls TSO/E to execute the REXX EXEC named ICHSPTCV.
	Input: The started procedures table report from step DSMON.
	Output: Resource definition statements for the STARTED class. The statements are placed into the data set pointed to by the OUTFILE DD statement.

See B.1, “ICHSPTCV Sample Output” on page 84 for a sample of the output from the REXX EXEC ICHSPTCV.

B.1 ICHSPTCV Sample Output

This topic provides a sample output from the REXX convert EXEC ICHSPTCV.

```
SETR GENERIC(STARTED)
RDEFINE STARTED PROCXYZ.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(NO) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED JES2.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED NET.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED NETO.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED TSO.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED IRRDPTAB.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED DFHSM.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED RMF.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED LLA.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED SMF.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED LOGREC.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED AOFAPPL.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED CNMAPROC.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED AOFASSI.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED CNMASSI.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED ISPFLMF.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(NO) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED UCBMOUNT.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED SAMON.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED DVGSTQH.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(YES) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED RACF.* STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(NO) TRUSTED(NO) TRACE(NO) )
RDEFINE STARTED ** STDATA( USER(STC) GROUP(SYS1) PRIVILEGED(NO) TRUSTED(NO) TRACE(YES) )
```

The profile names are named as follows :

```
MEMBERNAME.JOBNAME
```

Where MEMBERNAME is the real name of the member in the data set that contains the procedure, and JOBNAME is the name you can assign to that started task by starting it with a command like “START MEM1,JOBNAME=MYPROC.”

B.2 Executing the Generated Commands

The following JCL can be used to execute the RACF commands generated by the REXX EXEC ICHSPTCV:

```
//STARTED  EXEC PGM=IKJEFT01
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD DUMMY
//INFILE   DD DSN=&&TEMP,DISP=(OLD,DELETE)
//SYSTSIN  DD *
  EXEC 'userid.STARTED.DEFS'
/*
```

Notes:

1. Please note that the generated commands do not activate the RACF class STARTED. You must activate this class explicitly by executing the following commands:

   ```
   SETROPTS CLASSACT(STARTED)
   SETROPTS RACLIST(STARTED) REFRESH
   ```

2. The STARTED class must be SETROPTS RACLISTed.

3. When you specify TRACE(YES) for an entry in the STARTED class, RACF issues message IRR812I when that entry is used to assign an ID to a started task. The following examples show the result of the TRACE(YES) keyword when a procedure named ALLDEL is started:

```
S ALLDEL

IRR812I PROFILE ALLDEL.* (G) IN THE STARTED CLASS WAS USED
        TO START ALLDEL WITH JOBNAME ALLDEL.

S ALLDEL,JOBNAME=TESTJOB

IRR812I PROFILE ALLDEL.* (G) IN THE STARTED CLASS WAS USED
        TO START ALLDEL WITH JOBNAME TESTJOB.
```

Appendix C. Database Range Table - Sample JCL

This appendix provides a sample SMP/E job to implement the RACF *database range table*.

```
//ASM1    EXEC PGM=IEV90,PARM='DECK,NOOBJECT'
//SYSLIB   DD DISP=SHR,DSN=SYS1.AGENLIB
//         DD DISP=SHR,DSN=SYS1.AMODGEN
//SYSUT1   DD UNIT=VIO,SPACE=(CYL,(20,5))
//SYSUT2   DD UNIT=VIO,SPACE=(CYL,(10,1))
//SYSUT3   DD UNIT=VIO,SPACE=(CYL,(2,1))
//SYSPRINT DD SYSOUT=*
//SYSPUNCH DD DSN=&&TEMP,DISP=(,PASS),
//            SPACE=(CYL,(5,1)),UNIT=VIO
//SYSIN    DD *
         PUNCH '++USERMOD (RA00001) REWORK(1994200) .'
         PUNCH '++ VER (Z038) FMID(HRF2210).'
         PUNCH '++ MOD (ICHRRNG) DISTLIB(AOSBN) LMOD(ICHRRNG).'
ICHRRNG  CSECT
ICHRRNG  AMODE  31
ICHRRNG  RMODE  24
*
         DC    F'1'                    ONE RANGE
*
         DC    44X'00'                 RANGE START - ZEROS
         DC    AL1(1)                  DATABASE NUMBER
         END
/*
//*
//SMP2    EXEC PGM=GIMSMP,COND=(0,NE)
//SMPCSI   DD DISP=SHR,DSN=TOT.SMP.GLOBAL.CSI
//SMPHOLD  DD DUMMY
//SMPPTFIN DD DSN=&&TEMP,DISP=(OLD,DELETE)
//LPALIB   DD DISP=SHR,DSN=SYS1.LPALIB,
//            UNIT=SYSALLDA,VOL=SER=RES02A
//SMPCNTL  DD *
  SET BDY(TGTM02A) .
    UCLIN .
      REP LMOD(ICHRRNG) SYSLIB(LPALIB) .
    ENDUCL .
  SET BDY(GLOBAL) .
    RECEIVE S(RA00001) SYSMODS .
  SET BDY(TGTM02A) .
    APPLY REDO S(RA00001) .
/*
```

The REWORK operand (which contains up to 8 numeric characters) allows an updated SYSMOD to be automatically RE-RECEIVED (without a REJECT) as long as it is more recent than the version of the SYSMOD which is already RECEIVED.

Appendix D. Database Name Table - Sample JCL

This appendix provides a sample SMP/E job to implement the RACF *database name table*.

```
//ASM1    EXEC PGM=IEV90,PARM='DECK,NOOBJECT'
//SYSLIB   DD DISP=SHR,DSN=SYS1.AGENLIB
//         DD DISP=SHR,DSN=SYS1.AMODGEN
//SYSUT1   DD UNIT=VIO,SPACE=(CYL,(20,5))
//SYSUT2   DD UNIT=VIO,SPACE=(CYL,(10,1))
//SYSUT3   DD UNIT=VIO,SPACE=(CYL,(2,1))
//SYSPRINT DD SYSOUT=*
//SYSPUNCH DD DSN=&&TEMP,DISP=(,PASS),
//            SPACE=(TRK,(1,1)),UNIT=VIO
//SYSIN    DD *
         PUNCH '++ USERMOD (RA00002) REWORK(1994200) .'
         PUNCH '++ VER (Z038) FMID(HRF2210).'
         PUNCH '++ MOD (ICHRDSNT) DISTLIB(AOSBN) LMOD(ICHRDSNT).'
ICHRDSNT TITLE 'RACF DATA SET NAME TABLE'
ICHRDSNT CSECT  ,
ICHRDSNT AMODE  31
ICHRDSNT RMODE  24
*
         DC    AL1(1)                 ONE PRIMARY/BACKUP PAIR
         DC    CL44'SYS1.RACFESA'     DSN PRIMARY
         DC    CL44'SYS1.RACF.BKUP1'  DSN BACKUP
         DC    AL1(255)               RESIDENT DATA BLOCKS
         DC    B'10001000'            FLAG BYTE
         END
/*
//*
//SMP2    EXEC PGM=GIMSMP,COND=(0,NE)
//SMPCSI   DD DISP=SHR,DSN=TOT.SMP.GLOBAL.CSI
//SMPHOLD  DD DUMMY
//SMPPTFIN DD DSN=&&TEMP,DISP=(OLD,DELETE)
//LINKLIB  DD DISP=SHR,DSN=SYS1.LINKLIB,
//            UNIT=SYSALLDA,VOL=SER=RES02A
//SMPCNTL  DD *
  SET BDY(TGTM02A) .
    UCLIN .
      REP LMOD(ICHRDSNT) SYSLIB(LINKLIB) .
    ENDUCL .
  SET BDY(GLOBAL) .
    RECEIVE S(RA00002) SYSMODS .
  SET BDY(TGTM02A) .
    APPLY REDO S(RA00002) .
/*
```

Description of the flag byte bits :

Bit 0 Duplicate updates, but no statistics, to the backup database.

Bit 1 Duplicate statistical updates to the backup database.

Bit 2 Not used.

Bit 3 Not used.

Bit 4 Enable RACF sysplex communication.

Bit 5 Enable RACF sysplex data sharing (requires RACF sysplex communication enabled).

If multiple primary/backup database pairs are defined, the bits 4 and 5 are only relevant for the first entry of the database name table, RACF ignores these bits for subsequent entries.

Bit 6 Not used.

Bit 7 Not used.

Example: A flag byte setting of B′10001000′ will duplicate all updates except statistics to the backup database (bit 0) and enables RACF sysplex communication (bit 4).

Appendix E. Started Procedures Table - Sample JCL

This appendix provides a sample SMP/E job to implement a *started procedures table*.

```
//ASM1     EXEC PGM=IEV90,PARM='DECK,NOOBJECT'
//SYSLIB   DD DISP=SHR,DSN=SYS1.AGENLIB
//         DD DISP=SHR,DSN=SYS1.AMODGEN
//SYSUT1   DD UNIT=VIO,SPACE=(CYL,(20,5))
//SYSUT2   DD UNIT=VIO,SPACE=(CYL,(10,1))
//SYSUT3   DD UNIT=VIO,SPACE=(CYL,(2,1))
//SYSPRINT DD SYSOUT=*
//SYSPUNCH DD DSN=&&TEMP,DISP=(,PASS),
//            SPACE=(TRK,(1,1)),UNIT=VIO
//SYSIN    DD *
         PUNCH '++USERMOD (RA00003) REWORK(1994200).'
         PUNCH '++ VER (Z038) FMID(HRF2210).'
         PUNCH '++ MOD (ICHRIN03) DISTLIB(AOSBN) LMOD(ICHRIN03).'
ICHRIN03 TITLE 'RACF STARTED PROCEDURES TABLE'
ICHRIN03 CSECT
ICHRIN03 AMODE 31
ICHRIN03 RMODE ANY
         DC    AL2(ENTRIES+X'8000') NUMBER OF 32-BYTE ENTRIES
FIRST    DS    0H
         DC    CL8'procname'        PROCEDURE NAME
         DC    CL8'user    '        USER NAME
         DC    CL8'group   '        GROUP NAME
         DC    XL1'00'              FLAG BYTE
         DC    XL7'00'              RESERVED
         DC    CL8'*       '        GENERIC ENTRY
         DC    CL8'user    '        USER NAME
         DC    CL8'group   '        GROUP NAME
         DC    XL1'00'              FLAG BYTE
         DC    XL7'00'              RESERVED
LAST     DS    0H
ENTLEN   EQU   32                   ENTRY LENGTH
TOTLEN   EQU   LAST-FIRST           TOTAL LENGTH
ENTRIES  EQU   TOTLEN/ENTLEN        NUMBER OF ENTRIES
         END   ICHRIN03
/*
//SMP2    EXEC PGM=GIMSMP
//SMPCSI   DD DISP=SHR,DSN=TOT.SMP.GLOBAL.CSI
//SMPHOLD  DD DUMMY
//SMPPTFIN DD DSN=&&TEMP,DISP=(OLD,DELETE)
//LPALIB   DD DISP=SHR,DSN=SYS1.LPALIB,
//            UNIT=SYSALLDA,VOL=SER=RES02A
//SMPCNTL  DD *
  SET BDY(TGTM02A) .
    UCLIN .
      REP LMOD(ICHRIN03) SYSLIB(LPALIB) .
    ENDUCL .
  SET BDY(GLOBAL) .
    RECEIVE S(RA00003) SYSMODS .
  SET BDY(TGTM02A) .
    APPLY REDO S(RA00003) .
/*
```

Notes:

1. Be sure to define the following procedure names:

   IRRDPTAB RACF Dynamic Parsing.

   RACF The RACF subsystem address space.

   - If you code a generic entry, it must be the last one in the table.

2. Description of the flag byte bits:

   Bit 0 If set to on, it indicates that this entry has the PRIVILEGED attribute.

   Bit 1 If set to on, it indicates that this entry has the TRUSTED attribute.

   If both bit 0 and bit 1 are set on, the PRIVILEGED attribute overrides the TRUSTED attribute and no RACF auditing is done.

   Bits 2-7 Must be 0.

Appendix F. Administrative Data Utility - Sample JCL

This appendix provides a sample job to define the various cache structures in a coupling facility.

```
//ADU      EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//SYSIN    DD *
     DATA TYPE(CFRM) REPORT(YES)

     DEFINE POLICY NAME(CFRM01) REPLACE(YES)

        CF NAME(CF01)
           TYPE(009672)
           MFG(IBM)
           PLANT(02)
           SEQUENCE(000000040104)
           PARTITION(1)
           CPCID(00)
           DUMPSPACE(2000)

        CF NAME(CF02)
           TYPE(009672)
           MFG(IBM)
           PLANT(02)
           SEQUENCE(000000040104)
           PARTITION(1)
           CPCID(01)
           DUMPSPACE(2000)

        STRUCTURE NAME(IXC1_GRS)
           SIZE(1024)
           PREFLIST(CF01,CF02)

        STRUCTURE NAME(IXC1_DEFAULT)
           SIZE(4096)
           PREFLIST(CF02,CF01)

        STRUCTURE NAME(IXC_CICS)
           SIZE(4096)
           PREFLIST(CF02,CF01)

        STRUCTURE NAME(IRRXCF00_P001)
           SIZE(1644)
           PREFLIST(CF01,CF02)

        STRUCTURE NAME(IRRXCF00_B001)
           SIZE(329)
           PREFLIST(CF02,CF01)
           EXCLLIST(IRRXCF00_P001)
/*
```

F.1 Explanations

DATA TYPE(CFRM) REPORT(YES)

Indicates that the couple data set, which contains the CFRM data, is to be updated.

DEFINE POLICY NAME(CFRM01) REPLACE(YES)

Specifies the beginning of the definition for a new policy. A policy can only be defined in its entirety, so the *complete* policy must always be specified. The keyword REPLACE(YES) specifies that an eventually already existing policy with the same name will unconditionally be replaced.

This statement was not modified.

CF NAME(CF01) ...

Defines a coupling facility within the named policy.

This statement was not modified.

CF NAME(CF02) ...

Defines a second coupling facility within the named policy.

This statement was not modified.

STRUCTURE NAME(IXC1_GRS) ...

Defines a structure within the named policy.

This statement was not modified.

STRUCTURE NAME(IXC1_DEFAULT) ...

Defines a structure within the named policy.

This statement was not modified.

STRUCTURE NAME(IXC_CICS) ...

Defines a structure within the named policy.

This statement was not modified.

STRUCTURE NAME(IRRXCF00_P001) ...

This statement was added in this JCL run and defines the structure for the primary RACF 2.1 database.

The system will try to allocate this structure in the first coupling facility specified in the preference list (PREFLIST) parameter (CF01) and may also take the next specified CF in case of problems.

STRUCTURE NAME(IRRXCF00_B001) ...

This statement was added in this JCL run and defines the structure for the backup RACF 2.1 database.

The system will try to allocate this structure in the first coupling facility specified in the preference list (PREFLIST) parameter (in this case CF02) and may go further down the list in case of problems.

The exclusion list specifies a list of (up to 8) structure names with which this structure should possibly not share the same coupling facility. In other words, when structure IRRXCF00_P001 is allocated on CF01, the system will try to allocate the structure IRRXCF00_B001 on a coupling facility different from that, in this case on CF02.

Appendix G. Dynamic RACF Exit - Sample Code

```
ICHRIX01 TITLE 'ICHRIX01 DYNAMIC EXIT STUB'                              00010000
         SPACE 8                                                         00020000
         PRINT  ON                                                       00030000
         EJECT                                                           00040000
*********************************************************************** 00050000
*****   SAMPLE ICHRIX01                                             ***** 00060000
*****                                                               ***** 00070000
***** Function:                                                      ***** 00080000
*****                                                               ***** 00090000
*****   This exit uses the CSVDYNEX interface to call EXIT          ***** 00100000
*****   DYN.ICHRIX01.                                               ***** 00110000
*****                                                               ***** 00120000
*****   In SYS1.PARMLIB(PROGxx) we have defined an exit module      ***** 00130000
*****   called LOCRIX01 that relates to DYN.ICHRIX01.               ***** 00140000
*****                                                               ***** 00150000
*****   This gives us the possibility to use LOCRIX01 as a          ***** 00160000
*****   dynamic ICHRIX01 exit for RACF.                             ***** 00170000
*****                                                               ***** 00180000
*****   To inactivate LOCRIX01 we issue:                            ***** 00190000
*****   SETPROG EXIT,MODIFY,EXITNAME=DYN.ICHRIX01,                  ***** 00200000
*****           MODNAME=LOCRIX01,STATE=INACTIVE                     ***** 00210000
*****                                                               ***** 00220000
*****   To activate LOCRIX01 we issue:                              ***** 00230000
*****   SETPROG EXIT,MODIFY,EXITNAME=DYN.ICHRIX01,                  ***** 00240000
*****           MODNAME=LOCRIX01,STATE=ACTIVE                       ***** 00250000
*****                                                               ***** 00260000
*****   To activate new version of LOCRIX01 we issue:               ***** 00270000
*****   SETPROG EXIT,DELETE,EXITNAME=DYN.ICHRIX01,                  ***** 00280000
*****           MODNAME=LOCRIX01                                    ***** 00290000
*****   SETPROG EXIT,ADD,EXITNAME=DYN.ICHRIX01,                     ***** 00300000
*****           MODNAME=LOCRIX01,STATE=ACTIVE                       ***** 00310000
*****   (assume that a new LOCRIX01 was refreshed in LLA)           ***** 00320000
*****                                                               ***** 00330000
***** Logic:                                                        ***** 00340000
*****                                                               ***** 00350000
*****      QUERY DYN.ICHRIX01                                       ***** 00360000
*****          if not defined or only defined IMPLICIT then do      ***** 00370000
*****            DEFINE DYN.ICHRIX01                                ***** 00380000
*****          end                                                  ***** 00390000
*****      CALL DYN.ICHRIX01                                        ***** 00400000
*****          if exit module is active then do                     ***** 00410000
*****            the exit modul executes and returns                ***** 00420000
*****          end                                                  ***** 00430000
*****      END                                                      ***** 00440000
*****                                                               ***** 00450000
***** Prereq:                                                       ***** 00460000
*****                                                               ***** 00470000
*****    This must be defined in SYS1.PARMLIB(PROGxx):              ***** 00480000
*****                                                               ***** 00490000
*****    EXIT ADD                                                   ***** 00500000
*****         EXITNAME(DYN.ICHRIX01)                                ***** 00510000
*****         MODNAME(LOCRIX01)                                     ***** 00520000
*****                                                               ***** 00530000
*****    The LOCRIX01 module must be in the LNKLST concatenation    ***** 00540000
*****                                                               ***** 00550000
***** Customization:                                                ***** 00560000
*****                                                               ***** 00570000
*****    Some installations might want to add code, that covers     ***** 00580000
*****    all return and reason codes for every CSVDYNEX call.       ***** 00590000
*****                                                               ***** 00600000
*****    Some additional coding might be needed if DYN.ICHRIX01     ***** 00610000
```

```
*****    has several modules defined.                          ***** 00620000
*****                                                          ***** 00630000
***** Author: Erik Pauner    Date: July 22 1994                ***** 00640000
********************************************************************** 00660000
R0       EQU   0                                                     00670000
R1       EQU   1                                                     00680000
R2       EQU   2                                                     00690000
R3       EQU   3                                                     00700000
R4       EQU   4                                                     00710000
R5       EQU   5                                                     00720000
R6       EQU   6                                                     00730000
R7       EQU   7                                                     00740000
R8       EQU   8                                                     00750000
R9       EQU   9                                                     00760000
R10      EQU   10                                                    00770000
R11      EQU   11                                                    00780000
R12      EQU   12                                                    00790000
R13      EQU   13                                                    00800000
R14      EQU   14                                                    00810000
R15      EQU   15                                                    00820000
ICHRIX01 CSECT                                                       00830000
ICHRIX01 AMODE 31                                                    00840000
ICHRIX01 RMODE ANY                                                   00850000
         SPACE 3                                                     00860000
********************************************************************** 00870000
*****    SETUP SAVEAREA AND SAVE CALLER'S REGISTER             ***** 00880000
********************************************************************** 00890000
         SAVE  (14,12),,ICHRIX01..&SYSDATE                           00900000
         SPACE 1                                                     00910000
         LR    R12,R15                 SETUP OUR BASE                00920000
         SPACE 1                                                     00930000
         USING ICHRIX01,R12             TELL ASSEMBLER              00940000
         SPACE 1                                                     00950000
         LA    R0,WSLENGTH             GET LENGTH OF WORKING STORAGE 00960000
         GETMAIN R,LV=(0)              GETMAIN WORKING STORAGE       00970000
         SPACE 1                                                     00980000
         LR    R4,R1                   ADDRESS OF WS TO R4           00990000
         LR    R6,R1                   ADDRESS OF WS TO R6           01000000
         LA    R5,WSLENGTH             GET LENGTH OF WORKING STORAGE 01010000
         SLR   R7,R7                   CLEAR PATTERN REGISTER        01020000
         MVCL  R4,R6                   CLEAR WORKING STORAGE         01030000
         SPACE 1                                                     01040000
         ST    R1,8(,R13)              SAVE ADDR OF NEW SAVE IN OLD  01050000
         ST    R13,4(,R1)              SAVE ADDR OF OLD SAVE IN NEW  01060000
         LR    R13,R1                  SETUP NEW SAVE/WS             01070000
         SPACE 1                                                     01080000
         USING LSAVEAREA,R13           TELL ASSEMBLER                01090000
         L     R1,WSSAVO13             GET A(PREVIOUS SAVE           01100000
         L     R1,WSSAVE1-WSSAVE(,R1)  GET PARAMETER REGISTER        01110000
         EJECT                                                       01120000
*                                                                    01130000
* QUERY EXIT                                                         01140000
*                                                                    01150000
         CSVDYNEX  REQUEST=QUERY,EXITNAME=LEX,QTYPE=CALL,           C01160000
               RETCODE=LRETCODE,RSNCODE=LRSNCODE,WORKAREA=WSDYNQRY, C01170000
               MF=(E,WSDYNEX)                                        01180000
         NC    LRSNCODE,=AL4(CSVDYNEXRSNCODEMASK) AND OFF EXTRA BITS 01190000
         CLC   LRETCODE,=AL4(CSVDYNEXRC_OK)                          01200000
         BE    CALLROUT                  EXIT AND MODULE AVAIABLE    01210000
         CLC   LRETCODE,=AL4(CSVDYNEXRC_WARN)                        01220000
         BNE   ERRMSG1                                               01230000
         CLC   LRSNCODE,=AL4(CSVDYNEXRSNNOMODULES)                   01240000
         BE    RETURNOO                  NO MODULES IN EXIT ... END  01250000
         CLC   LRSNCODE,=AL4(CSVDYNEXRSNQUERYNOTFOUND)               01260000
         BE    DEFROUT             EXIT NOT DEFINED ..... GO DEFINE IT 01270000
         CLC   LRSNCODE,=AL4(CSVDYNEXRSNIMPLICITLYDEFINED)           01280000
```

```
         BE    DEFROUT             EXIT ONLY IMPLICIT ... GO DEFINE IT  01290000
         B     ERRMSG2                                                  01300000
*                                                                       01310000
* DEFINE EXIT                                                          01320000
*                                                                       01330000
DEFROUT  XC    LRSNCODE,LRSNCODE                                        01340000
         XC    LRETCODE,LRETCODE                                        01350000
         CSVDYNEX  REQUEST=DEFINE,EXITNAME=LEX,AMODE=31,PERSIST=IPL,   C01360000
               FASTPATH=NO,RETCODE=LRETCODE,RSNCODE=LRSNCODE,          C01370000
               MF=(E,WSDYNEX)                                           01380000
         CLC   LRETCODE,=AL4(CSVDYNEXRC_OK)                             01390000
         BE    CALLROUT                    ONLY RC=00 ACCEPTED          01400000
         B     ERRMSG3                                                  01410000
*                                                                       01420000
* CALL EXIT MODULE                                                     01430000
*                                                                       01440000
CALLROUT MVC   LRUBBITS,=X'40040000'   PASS R1 AND R13 TO EXIT MOD      01450000
         XC    LRSNCODE,LRSNCODE                                        01460000
         XC    LRETCODE,LRETCODE                                        01470000
         L     R10,WSSAV013                                             01480000
         ST    R13,LRUBR13             STORE OUR SAVE AREA FOR EXIT MOD 01490000
         LA    R10,20(,R10)                                             01500000
         MVC   LRUBR1(L'LRUBR1),4(R10) STORE OLD R1 FOR EXIT            01510000
         XC    LNEXTTOKEN,LNEXTTOKEN    INITIALIZE NEXT TOKEN           01520000
*                                                                       01530000
         CSVDYNEX  REQUEST=CALL,EXITNAME=LEX,FASTPATH=NO,              C01540000
               NEXTTOKEN=LNEXTTOKEN,RUB=LRUB,RETINFO=LAST,             C01550000
               RETAREA=LRETAREA,RETLEN==AL4(RETALEN),                  C01560000
               RETCODE=LRETCODE,RSNCODE=LRSNCODE,                      C01570000
               MF=(E,WSDYNEX)                                           01580000
         NC    LRSNCODE,=AL4(CSVDYNEXRSNCODEMASK) AND OFF EXTRA BITS    01590000
         CLC   LRETCODE,=AL4(CSVDYNEXRC_OK)                             01600000
         BE    SETRC                       THE END, GET R15 FROM MODULE 01610000
         CLC   LRETCODE,=AL4(CSVDYNEXRC_WARN)                           01620000
         BE    SAYRC4                                                   01630000
         CLC   LRETCODE,=AL4(CSVDYNEXRC_INVPARM)                        01640000
         BE    SAYRC8                                                   01650000
         CLC   LRETCODE,=AL4(CSVDYNEXRC_ENV)                            01660000
         BE    SAYRCC                                                   01670000
         CLC   LRETCODE,=AL4(CSVDYNEXRC_COMPERROR)                      01680000
         BE    SAYRC10                                                  01690000
*                                                                       01700000
         WTO   'DYN.ICHRIX01 RC>10'                                     01710000
         B     RETURN00                                                 01720000
*                                                                       01730000
SAYRC4   CLC   LRSNCODE,=AL4(CSVDYNEXRSNNOMODULES)                      01740000
         BE    RETURN00            NO EXIT MODULE, SO NO MSG ISSUED     01750000
*                                   THIS EXIT IS JUST DUMMY              01760000
         WTO   'DYN.ICHRIX01 RC=04'                                     01770000
         B     RETURN00                                                 01780000
SAYRC8   WTO   'DYN.ICHRIX01 RC=08'                                     01790000
         B     RETURN00                                                 01800000
SAYRCC   WTO   'DYN.ICHRIX01 RC=0C'                                     01810000
         B     RETURN00                                                 01820000
SAYRC10  WTO   'DYN.ICHRIX01 RC=10'                                     01830000
         B     RETURN00                                                 01840000
ERRMSG1  WTO   'DYN.ICHRIX01 QUERY RC>04'                               01850000
         B     RETURN00                                                 01860000
ERRMSG2  WTO   'DYN.ICHRIX01 QUERY RC=04 AND NOT OK'                    01870000
         B     RETURN00                                                 01880000
ERRMSG3  WTO   'DYN.ICHRIX01 DEFINE NOT OK, RC>00'                      01890000
         B     RETURN00                                                 01900000
ERRMSG4  WTO   'DYN.ICHRIX01 MODULE CALLED ABENDED'                     01910000
         B     RETURN00                                                 01920000
SETRC    LA    R2,LRETAREA                                              01930000
         USING EXRET,R2                                                 01940000
```

```
         TM    EXRETFLAGS,EXRETABEND      DID MODULE ABEND ?           01950000
         BNZ   ERRMSG4                                                 01960000
         L     R3,EXRETCODE       INSERT RETURN CODE FROM EXIT MODULE  01970000
         B     RETURN             RETURN TO FREEMAIN, R3 CONTAINS RC   01980000
RETURN00 SLR   R3,R3              RESET R3                             01990000
RETURN   DS    0H                                                      02000000
         SPACE 1                                                       02010000
         LA    R0,WSLENGTH             GET LENGTH OF WS                02020000
         LR    R1,R13                  ADDRESS OF WS TO R1             02030000
         L     R13,WSSAV013            GET ADDRESS OF PREVIOUS SAVE    02040000
         SPACE 1                                                       02050000
         DROP  R13                     TELL ASSEMBLER                  02060000
         SPACE 1                                                       02070000
         FREEMAIN R,LV=(0),A=(1)       FREE WORKING STORAGE            02080000
         SPACE 1                                                       02090000
         LR    R15,R3                  SET RETURN CODE                 02100000
         RETURN (14,12),RC=(15)        RETURN TO CALLER                02110000
         EJECT                                                         02120000
*************************************************************************02130000
*****          C O N S T A N T S                                 *****  02150000
*************************************************************************02170000
         SPACE 3                                                       02180000
         EJECT                                                         02190000
LEX      DC    CL16'DYN.ICHRIX01'                                      02200000
         LTORG                                                         02210000
*************************************************************************02220000
*****          SAVE AREA LAYOUT                                  *****  02240000
*************************************************************************02260000
         SPACE 1                                                       02270000
LSAVEAREA DSECT                         WORKING STORAGE DSECT         02280000
         SPACE 1                                                       02290000
WSSAVE   DS    F                       FIRST WORD OF SAVEAREA        02300000
WSSAV013 DS    F                       POINTER TO PREVIOUS SAVEAREA  02310000
WSSAVE13 DS    F                       SAVE FOR R13                  02320000
WSSAVE14 DS    F                       SAVE FOR R14                  02330000
WSSAVE15 DS    F                       SAVE FOR R15                  02340000
WSSAVE0  DS    F                       SAVE FOR R0                   02350000
WSSAVE1  DS    F                       SAVE FOR R1                   02360000
WSSAVE2  DS    F                       SAVE FOR R2                   02370000
WSSAVE3  DS    F                       SAVE FOR R3                   02380000
WSSAVE4  DS    F                       SAVE FOR R4                   02390000
WSSAVE5  DS    F                       SAVE FOR R5                   02400000
WSSAVE6  DS    F                       SAVE FOR R6                   02410000
WSSAVE7  DS    F                       SAVE FOR R7                   02420000
WSSAVE8  DS    F                       SAVE FOR R8                   02430000
WSSAVE9  DS    F                       SAVE FOR R9                   02440000
WSSAVE10 DS    F                       SAVE FOR R10                  02450000
WSSAVE11 DS    F                       SAVE FOR R11                  02460000
WSSAVE12 DS    F                       SAVE FOR R12                  02470000
DYNMODEL CSVDYNEX MF=(L,WSDYNEX)                                      02480000
WSDYNQRY DS    CL512                                                  02490000
RETALEN  EQU   L'EXRETHDR+L'EXRETINFO                                 02500000
LRETAREA DS    (RETALEN)CL1                                           02510000
LNEXTTOKEN DS  D                                                      02520000
LRETCODE DS    F                                                      02530000
LRSNCODE DS    F                                                      02540000
LRUB     DS    0XL12                                                  02550000
LRUBBITS DS    BL.32                                                  02560000
LRUBR1   DS    A                                                      02570000
LRUBR13  DS    A                                                      02580000
WSLENGTH EQU   *-LSAVEAREA                                            02590000
         PRINT OFF                                                    02600000
         EJECT                                                        02610000
         CSVEXRET                                                     02620000
         END   ICHRIX01                                               02630000
```

Appendix H. Source Code for a Demonstration Program

This appendix provides the source code of the program that is used to demonstrate the RACLIST enhancements. The logic of the program is discussed in Chapter 5, “RACLIST Enhancements” on page 39.

The program has to be linked AC-1 and must reside in an APF authorized library.

```
TESTDS   TITLE 'Demonstrate the RACLIST  enhancements in RACF V2R1'
***--PROPRIETARY_STATEMENT-----------------------------------------***
***                                                                 ***
***  (C) COPYRIGHT IBM CORP. 1994                                   ***
***  (C) COPYRIGHT IBM DEUTSCHLAND                                  ***
***                SYSTEME UND NETZE GMBH 1994                      ***
***  SEE COPYRIGHT INSTRUCTIONS                                     ***
***  PROGRAM PROPERTY OF IBM                                        ***
***                                                                 ***
***--END_OF_PROPRIETARY_STATEMENT---------------------------------***
***                                                                 ***
***      Program:   TESTDS to test RACF VERSION 2, RELEASE 1        ***
***                                                                 ***
***      Author:    H. Martin Dzatkowski                            ***
***                 IBM Germany, Dept. 3331                         ***
***                 SN IP System Development MVS                    ***
***                 Am Keltenwald  1                                ***
***                 71139 Ehningen                                  ***
***                                                                 ***
***      Function:  The program reads information from the          ***
***                 RACLIST data space for the class FACILITY       ***
***                                                                 ***
***                 The class FACILITY is RACLISTed and has to      ***
***                 be refreshed to make the updates to the         ***
***                 profiles active                                 ***
***                                                                 ***
***                                                                 ***
***                 Tested RACROUTE requests are                    ***
***                                                                 ***
***                 EXTRACT,BRANCH=YES                              ***
***                 AUTH                                            ***
***                 FASTAUTH                                        ***
***                                                                 ***
***                 All tests can be performed without or           ***
***                 with a preceding LIST,GLOBAL=YES                ***
***                                                                 ***
***-----------------------------------------------------------------***
         SPACE 4
R0       EQU   0                   Parameter
R1       EQU   1                   Parameter
R2       EQU   2                   WorkReg
R3       EQU   3                   WorkReg
R4       EQU   4                   WorkReg, Base Register ACEE
R5       EQU   5                   WorkReg
R6       EQU   6                   WorkReg
R7       EQU   7                   WorkReg
R8       EQU   8                   Base Register Extracted Text
R9       EQU   9                   WorkReg, Counter
R10      EQU   10                  Counter
R11      EQU   11                  Base Register Parameter list
```

```
R12      EQU   12                Base Register TESTDS
R13      EQU   13                Base Register SaveArea
R14      EQU   14                Linkage
R15      EQU   15                Linkage
         EJECT
TESTDS   CSECT
TESTDS   AMODE 31
TESTDS   RMODE ANY
         SPACE 4
         USING TESTDS,R15        establish Base Register for Exit
         B     CRIGHT            BRANCH AROUND ID.
         SPACE 4
***------------------------------------------------------------***
***      Program Identifier and Copyright Statement            ***
***------------------------------------------------------------***
         DC    AL2(CRIGHT-*)
         DC    C'TESTDS  .'      Program Identifier
         DC    C'V01.R00 '
         DC    C'5740-XXH '      Program Number
         DC    C'&SYSDATE'       Assembly Date
         DC    C'&SYSTIME'       Assembly Time
         DS    0C                Security Eye Catcher
         DC    C'(C) COPYRIGHT IBM CORP. 1994, 1994. '
         DC    C'(C) COPYRIGHT IBM DEUTSCHLAND '
         DC    C'SYSTEME UND NETZE GMBH. 1994, 1994. '
         DC    C'PROGRAM PROPERTY OF IBM '
         DC    C'/CLASS=RESTRICTED MATERIAL OF IBM'
         DC    C'/OWN=SN IP MVS SSC'
         DC    C'/AUTHOR=H. MARTIN DZATKOWSKI '
         DC    C'/VNET=DEIBMDZA AT IBMMAIL   '
CRIGHT   DS    0H
         SPACE 4
         SAVE  (14,12)           SAVE REGISTERS
         LR    R12,R15           SET UP BASE
         DROP  R15
         USING TESTDS,R12        establish Base for Routine
         LR    R2,R1             save Addr. of Parameter List
         LR    R7,R13            save Addr. of old SaveArea
         L     R13,=A(SAVEAREA)  prepare addressability of Data
         USING SAVEAREA,R13
         ST    R7,SAVEAREA+4     store Addr. of old SaveArea
         ST    R13,8(R7)         store Addr. of new SaveArea
         SPACE 3
***------------------------------------------------------------***
***      Open the Input and Output Datasets                    ***
***------------------------------------------------------------***
         SPACE 1
         OPEN  (PRINT,OUTPUT)
         SPACE 3
***------------------------------------------------------------***
***      Prepare the analysis of the Parameters given          ***
***------------------------------------------------------------***
         SPACE 1
         L     R2,0(R2)          GET PARAMETER LIST
         LH    R3,0(R2)          GET PARAMETER COUNT
         LA    R2,2(R2)          GET PARAMETER ADDRESS
         LTR   R3,R3             Test, if any Parameter give ?
         BZ    NOTHING
         MVC   HEADER+13(50),EMPTY
```

```
         LR    R1,R3
         BCTR  R1,0
         EX    R1,*+4
         MVC   HEADER+13(*-*),0(R2)
         PUT   PRINT,HEADER        PRINT Header and Parameter Field
         PUT   PRINT,EMPTY
         EJECT
***--------------------------------------------------------------***
***      Parse thru the given Parameter list                     ***
***--------------------------------------------------------------***
PARSE    DS    0H
         LR    R11,R2              keep Parameter Addr. for set back
         SPACE 3
***--------------------------------------------------------------***
***      OWNER=UserID                                            ***
***--------------------------------------------------------------***
PARMUID  CLC   0(6,R2),=C'OWNER='
         BNE   PARMATTR
         LA    R4,LENUID
         LA    R5,VARUID
         LA    R6,6                Parameter Name Length
         LA    R8,8                maximal Value length
         B     READLONG
         SPACE 3
***--------------------------------------------------------------***
***      REQUEST=A/F/E possible: Auth, FastAuth, Extract         ***
***--------------------------------------------------------------***
PARMATTR CLC   0(8,R2),=C'REQUEST='
         BNE   PARMRACL
         LA    R5,VARATTR
         LA    R6,8                Parameter Name Length
         B     READONE
         SPACE 3
***--------------------------------------------------------------***
***      RACLIST=Y/N possible: yes, no                           ***
***--------------------------------------------------------------***
PARMRACL CLC   0(8,R2),=C'RACLIST='
         BNE   INVPARM
         LA    R5,VARRACL
         LA    R6,8
         B     READONE
         SPACE 3
***--------------------------------------------------------------***
***      Get the Value of one found Parameter                    ***
***--------------------------------------------------------------***
READLONG DS    0H
         SR    R3,R6
         BNP   NULVALUE
         AR    R2,R6
         CLI   0(R2),C','          Comma next char => no Value given
         BE    NULVALUE
         LR    R9,R2
         XR    R7,R7
NEXTCHAR CLI   0(R2),C','
         BE    LASTLONG
         LA    R2,1(R2)
         LA    R7,1(R7)
         BCT   R3,NEXTCHAR
LASTLONG DS    0H
```

```
         LTR   R7,R7
         BZ    NULVALUE
         CR    R7,R8
         BH    BIGVALUE
         STC   R7,0(,R4)
         BCTR  R7,0
         EX    R7,VALMOVE
         B     NEXTPARM
VALMOVE  MVC   0(,R5),0(R9)
         SPACE 3
***-------------------------------------------------------------***
***      Check Parameter of length 1                            ***
***-------------------------------------------------------------***
READONE  DS    0H
         SR    R3,R6
         BNP   NULVALUE
         AR    R2,R6
         CLI   0(R2),C',          Comma next char => no Value given
         BE    NULVALUE
         CH    R3,=H'1'           Last char => Value OK
         BE    MOVEPARM
         CLI   1(R2),C',          Comma not next char => Value to big
         BNE   BIGVALUE
MOVEPARM DS    0H
         MVC   0(1,R5),0(R2)
         BCTR  R3,0
         LA    R2,1(R2)
         SPACE 3
***-------------------------------------------------------------***
***      Find the next Parameter, if any left                   ***
***-------------------------------------------------------------***
NEXTPARM DS    0H
         LTR   R3,R3              Test more Parameter given ?
         BZ    GETLIST
         LA    R2,1(R2)
         BCTR  R3,0               remove comma
         LTR   R3,R3              Test more Parameter give ?
         BNZ   PARSE
         EJECT
***-------------------------------------------------------------***
***      First RACLIST all the profiles                         ***
***-------------------------------------------------------------***
         SPACE 1
GETLIST  DS    0H
         CLI   VARATTR,C'E'
         BNE   CHCKLIST           For REQUEST=EXTRACT UserID request
         CLI   LENUID,X'00'
         BE    NOOWNER            none given, finish with message
CHCKLIST DS    0H
         PUT   PRINT,VARHDR
         CLI   VARRACL,C'Y'
         BNE   CHCKREQ            Check, which Request to be issued
         MVC   VARROUT,=CL8'List,G=Y'
         LA    R5,WORKA           get Addr. of WorkArea
EXECLIST RACROUTE REQUEST=LIST,                                  *
               ENVIR=CREATE,                                     *
               RELEASE=2.1,                                      *
               GLOBAL=YES,                                       *
               CLASS=NAMFAC,                                     *
```

```
              WORKA=(R5),                                              *
              MF=S
         EJECT
***-------------------------------------------------------------------***
***      Get ReturnCodes and show them                                ***
***-------------------------------------------------------------------***
         SPACE 1
         ST    R15,SAFRCODE        save the SAF ReturnCode
         CVD   R15,DECIMAL#
         MVC   SAFRC,MASK#4
         ED    SAFRC,DECIMAL#+6
         L     R5,EXECLIST+4       get the RACF ReturnCode
         CVD   R5,DECIMAL#
         MVC   RACFRC,MASK#4
         ED    RACFRC,DECIMAL#+6
         L     R6,EXECLIST+8       get the RACF ReasonCode
         CVD   R6,DECIMAL#
         MVC   RACFREAS,MASK#4
         ED    RACFREAS,DECIMAL#+6
         TIME  DEC,DECTIME,LINKAGE=SYSTEM
         MVC   CHARTIME,MASK#TIM
         ED    CHARTIME,DECTIME
         MVC   VARHOUR,CHARTIME
         MVC   VARMIN,CHARTIME+2
         MVC   VARSEC,CHARTIME+4
         PUT   PRINT,VARDATA
         L     R15,SAFRCODE        get the SAF ReturnCode back
         LTR   R15,R15             completed successfully
         BNZ   FINISH
         EJECT
***-------------------------------------------------------------------***
***      Some preparation current for all RACF Exits                  ***
***-------------------------------------------------------------------***
         SPACE 1
CHCKREQ  DS    0H
         XR    R9,R9
         CLI   VARATTR,C'E'
         BE    GETEXTR             UserID from Parameter list used
***-------------------------------------------------------------------***
***      Get the UserID from the current ACEE                         ***
***-------------------------------------------------------------------***
         SPACE 1
         XR    R0,R0
         USING PSA,R0              establish Base for PSA
         L     R5,PSAAOLD          get Addr of ASCB
         USING ASCB,R5             establish Base for ASCB
         L     R4,PSATOLD          get Addr OF TCB
         USING TCB,R4              establish Base for TCB
         DROP  R0
         L     R2,TCBSENV          get address of ACEE
         DROP  R4
         LTR   R2,R2               ACEE address pointed to by TCB ?
         BNZ   CHCKACEE            address filled, test for ACEE
         SPACE 1
FROMASXB DS    0H
         LTR   R5,R5               ASCB address found ?
         BZ    CHCKAUTH            no ASCB given, no ACEE to find
         CLC   ASCBASCB,=CL4'ASCB'
         BNE   CHCKAUTH            not really ASCB, no ACEE found
```

```
         L     R2,ASCBASXB         get Addr of ASCB extension
         DROP  R5                  release addressing of ASCB
         LTR   R2,R2               ASXB address found ?
         BZ    CHCKAUTH            no ASXB given, no ACEE to find
         USING ASXB,R2             establish Base for ASCB
         CLC   ASXBASXB,=CL4'ASXB'
         BNE   CHCKAUTH            not really ASXB, no ACEE found
         L     R2,ASXBSENV         get address of ACEE
         LTR   R2,R2               ACEE address pointed to by ASXB ?
         BZ    CHCKAUTH            no ACEE to find
         DROP  R2                  release addressing of ASXB
CHCKACEE DS    0H
         USING ACEE,R2             establish Base Register of ACEE
         CLC   ACEEACEE,=CL4'ACEE'
         BNE   CHCKAUTH            not really an ACEE
         MVC   LENUID(9),ACEEUSER
         DROP  R2                  release addressing of ASXB
         SPACE 3
***-------------------------------------------------------------***
***      UserID from the ACEE used in this two cases            ***
***-------------------------------------------------------------***
         SPACE 1
CHCKAUTH DS    0H
         CLI   VARATTR,C'A'
         BE    GETAUTH
         XR    R10,R10             set counter for FASTAUTH Loop
         LH    R10,=H'10'          set the Count for first 10 Msgs
         CLI   VARATTR,C'F'
         BE    GETFAST
         MVC   IVTEXT+30(8),=C'REQUEST='
         MVC   IVTEXT+38(1),VARATTR
         CLI   VARATTR,C' '        Parameter REQUEST= not specified ?
         BNE   INVVALUE
         EJECT
***-------------------------------------------------------------***
***      Get the Access Authority Extracted                     ***
***-------------------------------------------------------------***
         SPACE 1
GETEXTR  DS    0H
         LA    R9,1(R9)
         MVC   VARROUT,=CL8'Extract'
         MVC   ENTITY,ENTDSPC      Profile to be checked
         LA    R5,WORKA            get Addr. of WorkArea
EXECEXTR RACROUTE REQUEST=EXTRACT,                                  *
               TYPE=EXTRACT,                                        *
               RELEASE=2.1,                                         *
               BRANCH=YES,                                          *
               SUBPOOL=2,                                           *
               CLASS=NAMFAC,                                        *
               ENTITY=ENTITY,                                       *
               FIELDS=FLDSFAC,                                      *
               WORKA=(R5),                                          *
               MF=S
         ST    R15,SAFRCODE        save the SAF ReturnCode
         LTR   R15,R15
         BNZ   SHOWEXTW            Invalid Profile, skip Analysis
         SPACE 1
         USING EXTWKEA,R8          Addressing of extended WorkArea
         ST    R1,EXTWADDR         save the Addr. for FreeMain
```

```
         LR    R8,R1               Base of EXTRACT WorkArea
         AH    R8,EXTWOFF          and add the Offset
         DROP  R8                  release
         USING EXTDATA,R8          Addressing of variable part
         CLC   VAROWNER,EXTOWNER   compare Owner and last shown
         BE    FREEEXTW
         MVC   VAROWNER,EXTOWNER   get the Owner and show it
         DROP  R8                  release
         EJECT
***------------------------------------------------------------------***
***      Show the Result of the last RACROUTE, if changed            ***
***------------------------------------------------------------------***
         SPACE 1
SHOWEXTW DS    0H
         L     R15,SAFRCODE        get the SAF ReturnCode
         CVD   R15,DECIMAL#
         MVC   SAFRC,MASK#4
         ED    SAFRC,DECIMAL#+6
         L     R5,EXECEXTR+4       get the RACF ReturnCode
         CVD   R5,DECIMAL#
         MVC   RACFRC,MASK#4
         ED    RACFRC,DECIMAL#+6
         L     R6,EXECEXTR+8       get the RACF ReasonCode
         CVD   R6,DECIMAL#
         MVC   RACFREAS,MASK#4
         ED    RACFREAS,DECIMAL#+6
         L     R15,SAFRCODE        get the SAF ReturnCode
         LTR   R15,R15
         BZ    EDITTIME            Following test for other RC
         CLC   ALLCODES,SAFRC
         BE    COMPGOAL
         MVC   ALLCODES,SAFRC
         MVC   VAROWNER,EMPTY      clear the output owner field
EDITTIME DS    0H
         TIME  DEC,DECTIME,LINKAGE=SYSTEM
         MVC   CHARTIME,MASK#TIM
         ED    CHARTIME,DECTIME
         MVC   VARHOUR,CHARTIME
         MVC   VARMIN,CHARTIME+2
         MVC   VARSEC,CHARTIME+4
         CVD   R9,DECIMAL#
         MVC   VARLEAP#,MASK#
         ED    VARLEAP#,DECIMAL#+4
         PUT   PRINT,VARDATA
         SPACE 3
***------------------------------------------------------------------***
***      Free the Storage obtained by RACROUTE,REQ=EXTRACT           ***
***------------------------------------------------------------------***
         SPACE 1
FREEEXTW DS    0H
         XR    R3,R3               CLEAR REGISTER
         ICM   R3,B'1111',EXTWADDR load Addr. of data area
         BZ    COMPGOAL            skip FreeMain
         XR    R2,R2               CLEAR REGISTER
         ST    R2,EXTWADDR         clear Addr. of data area again
         LH    R2,EXTWSP-EXTWKEA+2(,R3)  load SubPool and length
         FREEMAIN RC,A=(3),LV=(2),SP=2   free data area
         LTR   R15,R15
         BZ    COMPGOAL            Storage freed, go on
```

```
         PUT   PRINT,FREEERR       show, that FREEMAIN doesn't work
         B     LISTDEL             end this Program
         SPACE 1
COMPGOAL DS    OH
         CLC   VAROWNER,VARUID     compare delimiting condition
         BNZ   GETEXTR
         B     LISTDEL
         EJECT
***-------------------------------------------------------------***
***      Check the Authorization of the UserID                  ***
***-------------------------------------------------------------***
         SPACE 1
GETFAST  DS    OH
         LA    R9,1(R9)
         MVC   VARROUT,=CL8'FastAuth'
         MVC   VARREQU,=CL8'Control'
         MVC   ENTITY,ENTDSPC      Profile to be checked
         LA    R5,WORKA            get Addr. of WorkArea
         LA    R6,WKAREA           get Addr. of WK-Area
         MVC   WKAREA,EMPTY        clear the WK-Area
EXECFAST RACROUTE REQUEST=FASTAUTH,                                *
               RELEASE=2.1,                                        *
               CLASS=NAMFAC,                                       *
               ENTITY=ENTITY,                                      *
               ATTR=CONTROL,                                       *
               WORKA=(R5),                                         *
               WKAREA=(R6),                                        *
               MF=S
         EJECT
***-------------------------------------------------------------***
***      Get ReturnCodes and react related to the error message  ***
***-------------------------------------------------------------***
         SPACE 1
         ST    R15,SAFRCODE        save the SAF ReturnCode
         CVD   R15,DECIMAL#
         MVC   SAFRC,MASK#4
         ED    SAFRC,DECIMAL#+6
         L     R5,EXECFAST+4       get the RACF ReturnCode
         CVD   R5,DECIMAL#
         MVC   RACFRC,MASK#4
         ED    RACFRC,DECIMAL#+6
         L     R6,EXECFAST+8       get the RACF ReasonCode
         CVD   R6,DECIMAL#
         MVC   RACFREAS,MASK#4
         ED    RACFREAS,DECIMAL#+6
         CLC   ALLCODES,SAFRC
         BE    GETFAST
         MVC   ALLCODES,SAFRC
SHOWFAST DS    OH
         TIME  DEC,DECTIME,LINKAGE=SYSTEM
         MVC   CHARTIME,MASK#TIM
         ED    CHARTIME,DECTIME
         MVC   VARHOUR,CHARTIME
         MVC   VARMIN,CHARTIME+2
         MVC   VARSEC,CHARTIME+4
         CVD   R9,DECIMAL#
         MVC   VARLEAP#,MASK#
         ED    VARLEAP#,DECIMAL#+4
         L     R15,SAFRCODE        get the SAF ReturnCode back
```

```
         LTR   R15,R15             completed successfully
         BZ    GRANTED
         CH    R15,=H'4'           4 = DataSpace was deleted
         BE    NODEC
         MVC   VARAUTH,=CL8'NotAuth'
         B     SHOWIT
NODEC    DS    0H
         MVC   VARAUTH,=CL8'NoDecPos'
         B     SHOWIT
GRANTED  DS    0H
         MVC   VARAUTH,=CL8'Granted'
SHOWIT   DS    0H
         PUT   PRINT,VARDATA
         L     R15,SAFRCODE        get the SAF ReturnCode back
         LTR   R15,R15             completed successfully
         BNZ   GETFAST
         B     LISTDEL
         EJECT
***-------------------------------------------------------------***
***      Check the Authorization of the UserID                  ***
***-------------------------------------------------------------***
         SPACE 1
GETAUTH  DS    0H
         LA    R9,1(R9)
         MVC   VARROUT,=CL8'Auth'
         MVC   VARREQU,=CL8'Update'
         MVC   ENTITY,ENTDSPC      Profile to be checked
         LA    R5,WORKA            get Addr. of WorkArea
EXECAUTH RACROUTE REQUEST=AUTH,                                  *
               RELEASE=2.1,                                      *
               CLASS=CLFAC,                                      *
               ENTITY=(ENTITY),                                  *
               STATUS=ACCESS,                                    *
               MSGSUPP=YES,                                      *
               WORKA=(R5),                                       *
               MF=S
         EJECT
***-------------------------------------------------------------***
***      Get ReturnCodes and react related to the error message ***
***-------------------------------------------------------------***
         SPACE 1
         ST    R15,SAFRCODE        save the SAF ReturnCode
         CVD   R15,DECIMAL#
         MVC   SAFRC,MASK#4
         ED    SAFRC,DECIMAL#+6
         L     R5,EXECAUTH+4       get the RACF ReturnCode
         CVD   R5,DECIMAL#
         MVC   RACFRC,MASK#4
         ED    RACFRC,DECIMAL#+6
         L     R6,EXECAUTH+8       get the RACF ReasonCode
         CVD   R6,DECIMAL#
         MVC   RACFREAS,MASK#4
         ED    RACFREAS,DECIMAL#+6
         LTR   R15,R15             completed successfully ?
         BNZ   SHOWDOWN
         CH    R5,=H'20'           x'14' indicates STATUS+ACCESS
         BNE   SHOWDOWN
         CH    R6,LASTACC
         BE    GETAUTH
```

```
         STH   R6,LASTACC
         B     *+4(R6)             get access authority by ReasonCode
         B     NONE
         B     READ
         B     UPDATE
         B     CONTROL
         B     ALTER
SHOWDOWN DS    OH
         CLC   ALLCODES,SAFRC
         BE    GETAUTH
         MVC   ALLCODES,SAFRC
         MVC   VARAUTH,EMPTY       clear output authorization field
         B     SHOWACC
NONE     DS    OH
         MVC   VARAUTH,=CL8'NONE'
         B     SHOWACC
READ     DS    OH
         MVC   VARAUTH,=CL8'READ'
         B     SHOWACC
UPDATE   DS    OH
         MVC   VARAUTH,=CL8'UPDATE'
         B     SHOWACC
CONTROL  DS    OH
         MVC   VARAUTH,=CL8'CONTROL'
         B     SHOWACC
ALTER    DS    OH
         MVC   VARAUTH,=CL8'ALTER'
SHOWACC  DS    OH
         TIME  DEC,DECTIME,LINKAGE=SYSTEM
         MVC   CHARTIME,MASK#TIM
         ED    CHARTIME,DECTIME
         MVC   VARHOUR,CHARTIME
         MVC   VARMIN,CHARTIME+2
         MVC   VARSEC,CHARTIME+4
         CVD   R9,DECIMAL#
         MVC   VARLEAP#,MASK#
         ED    VARLEAP#,DECIMAL#+4
         PUT   PRINT,VARDATA
         CH    R6,=H'8'
         BNE   GETAUTH
         EJECT
***-------------------------------------------------------------***
***      Last release all RACLISTed profiles                    ***
***-------------------------------------------------------------***
         SPACE 1
LISTDEL  DS    OH
         CLI   VARRACL,C'Y'
         BNE   FINISH              Everything is done
         MVC   VARROUT,=CL8'ENV=DEL'
         LA    R5,WORKA            get Addr. of WorkArea
EXECDEL  RACROUTE REQUEST=LIST,                                  *
               ENVIR=DELETE,                                     *
               RELEASE=2.1,                                      *
               CLASS=NAMFAC,                                     *
               WORKA=(R5),                                       *
               MF=S
         EJECT
***-------------------------------------------------------------***
```

```
***      Get ReturnCodes and show them                              ***
***-----------------------------------------------------------------***
         SPACE 1
         CVD   R15,DECIMAL#
         MVC   SAFRC,MASK#4
         ED    SAFRC,DECIMAL#+6
         L     R5,EXECDEL+4        get the RACF ReturnCode
         CVD   R5,DECIMAL#
         MVC   RACFRC,MASK#4
         ED    RACFRC,DECIMAL#+6
         L     R6,EXECDEL+8        get the RACF ReasonCode
         CVD   R6,DECIMAL#
         MVC   RACFREAS,MASK#4
         ED    RACFREAS,DECIMAL#+6
         TIME  DEC,DECTIME,LINKAGE=SYSTEM
         MVC   CHARTIME,MASK#TIM
         ED    CHARTIME,DECTIME
         MVC   VARHOUR,CHARTIME
         MVC   VARMIN,CHARTIME+2
         MVC   VARSEC,CHARTIME+4
         PUT   PRINT,VARDATA
         SPACE 3
***-----------------------------------------------------------------***
***      End of the Program                                         ***
***-----------------------------------------------------------------***
         SPACE 1
FINISH   DS    0H
         PUT   PRINT,EMPTY
         PUT   PRINT,FINISHED
         B     CLOSINGS
         SPACE 1
SETRC40  DS    0H
         PUT   PRINT,NOTEXT        no test-run performed
         MVI   RETC+1,40           Set Return Code = 40
         SPACE 1
CLOSINGS DS    0H
         CLOSE PRINT              CLOSE DATA SET SYSPRINT
         LH    R15,RETC           LOAD RETURN CODE
         L     R13,SAVEAREA+4     RESTORE SAVE AREA ADDRESS
         RETURN (14,12),RC=(15)   RELOAD REGISTERS AND RETURN WITH RC
         EJECT
***-----------------------------------------------------------------***
***      An Error occurred, handle and exit the test                ***
***-----------------------------------------------------------------***
NOTHING  PUT   PRINT,HEADER        PRINT Header and Parameter Field
         PUT   PRINT,EMPTY         Not any Parameter given
         B     SETRC40
         SPACE 1
NOOWNER  PUT   PRINT,OWTEXT        Owner to be searched missing
         B     SETRC40
         SPACE 1
INVPARM  DS    0H                  An unknown Parameter found
         MVC   IPTEXT+20(10),0(R2)
         PUT   PRINT,IPTEXT        Show and leave Program then
         B     SETRC40
         SPACE 1
NULVALUE DS    0H
         XR    R7,R7
```

```
         SPACE 1
BIGVALUE DS    OH
         LR    R2,R11              Reset to last Parameter Name
         AR    R7,R6
         EX    R7,PARMMOVE
         SPACE 1
INVVALUE DS    OH                  An unknown Value of a Parameter set
         PUT   PRINT,IVTEXT        Show and leave Program then
         B     SETRC40
PARMMOVE MVC   IVTEXT+30(0),0(R2)
         B     SETRC40
         SPACE 3
***------------------------------------------------------------***
***      Output                                                ***
***------------------------------------------------------------***
         SPACE 1
PRINT    DCB   DDNAME=SYSPRINT,DSORG=PS,MACRF=(PM),                    *
               RECFM=FA,LRECL=121,BLKSIZE=121,BUFNO=1
         DROP  R12                Release Base of Code
         EJECT
***------------------------------------------------------------***
***      Literal Pool                                          ***
***------------------------------------------------------------***
         SPACE 1
         LTORG
         SPACE 3
***------------------------------------------------------------***
***      Definitions, Constants                                ***
***------------------------------------------------------------***
         SPACE 1
SAVEAREA CSECT
         DC    18F'0'              Save Area
         SPACE 3
SAFRCODE DC    F'0'                Return Code of SAF Calls
ALLCODES DC    CL14' '             Return Codes (SAF,RACF,Reason)
EXTWADDR DC    A(0)                Addr. of data area from RACROUTE
DECIMAL# DC    D'0'                Number Conversion
DECTIME  DS    CL16                Time and Date in Decimal
RETC     DC    H'0'                RETURN CODE
LASTACC  DC    H'255'              Keep Access Authority to compare
CHARTIME DS    CL9                 Time to be edited
MASK#    DC    X'4020202020202120'
MASK#TIM DC    X'2020202020202020'
MASK#4   DC    X'40202120'
         SPACE 3
CLFAC    DC    XL1'08'
NAMFAC   DC    CL8'FACILITY'       Class to be checked
         SPACE 1
ENTDSPC  DC    CL39'DATASPACE'     Profile to be checked
         SPACE 1
FLDSFAC  DC    AL4(2)
         DC    CL8'OWNER'          Owner of the Profile
         DC    CL8'UACC'           Universal Access Authority
         SPACE 3
***------------------------------------------------------------***
***      Messages                                              ***
***------------------------------------------------------------***
         SPACE 1
HEADER   DC    CL121'1Parameters: >>>>> non given <<<<<'
```

```
EMPTY    DC    CL121' '
FINISHED DC    CL121' Program ended with RC:      0.  All Work done'
IPTEXT   DC    CL121' Invalid Parameter:'
IVTEXT   DC    CL121' Invalid Value for Parameter:'
OWTEXT   DC    CL121' Owner to search for missing'
NOTEXT   DC    CL121' No Test Run performed'
FREEERR  DC    CL121' Problem on freeing RACF GETMAINed Storage'
         EJECT
***-------------------------------------------------------------***
***      Data Space Parameter List                              ***
***-------------------------------------------------------------***
         SPACE 1
VARHDR   DS    CL121
         ORG   VARHDR
         DC    CL9' Loop-Ct'
         DC    CL9' Time'
         DC    CL9' Request'
         DC    CL9' UserID'
         DC    CL9' Owner'
         DC    CL5'  SAF'
         DC    CL5' RACF'
         DC    CL5'  Rsn'
         DC    CL9' Acc-Auth'
         DC    CL9' Acc.Requ'
         DC    CL4' Req'
         DC    CL9' '
         DC    CL9' '
         DC    CL9' '
         DC    CL9' '
         DC    CL3' '
         SPACE 3
VARDATA  DS    CL121
         ORG   VARDATA
         DC    CL1' '
VARLEAP# DC    CL8' '
         DC    CL1' '
VARHOUR  DC    CL2' '
         DC    CL1'.'
VARMIN   DC    CL2' '
         DC    CL1'.'
VARSEC   DC    CL2' '
         DC    CL1' '
VARROUT  DC    CL8' '
LENUID   DC    XL1'0'              entered by Parameter
VARUID   DC    CL8' '              entered by Parameter
         DC    CL1' '
VAROWNER DC    CL8' '
         DC    CL1' '
SAFRC    DC    CL4' '
         DC    CL1' '
RACFRC   DC    CL4' '
         DC    CL1' '
RACFREAS DC    CL4' '
         DC    CL1' '
VARAUTH  DC    CL8' '
         DC    CL1' '
VARREQU  DC    CL8' '
         DC    CL1' '
VARATTR  DC    CL1' '              entered by Parameter
```

```
         DC    CL1' '
VARRACL  DC    CL1' '               entered by Parameter
VARFLD1  DC    CL8' '
         DC    CL1' '
VARFLD2  DC    CL8' '
         DC    CL1' '
VARFLD3  DC    CL8' '
         DC    CL1' '
VARFLD4  DC    CL8' '
         DC    CL4' '
         SPACE 3
***------------------------------------------------------------------***
***      WorkArea for the RACROUTE macros                            ***
***------------------------------------------------------------------***
         SPACE 1
WKAREA   DS    16F
         SPACE 1
WORKA    DS    CL512
         SPACE 1
ENTITY   DS    CL39
         SPACE 3
***------------------------------------------------------------------***
***      Layout of the returned Data from RACROUTE=EXTRACT           ***
***------------------------------------------------------------------***
         SPACE 1
EXTDATA  DSECT                       return result of RACROUTE-EXTRACT
         DS    CL4
EXTOWNER DS    CL8
         DS    CL4
EXTUACC  DS    X
         PRINT OFF
         EJECT
         IRRPRXTW
         EJECT
         IKJTCB
         EJECT
         IHAACEE
         EJECT
         CVT      LIST=YES,DSECT=YES
         EJECT
         IHAPSA   LIST=YES,DSECT=YES
         EJECT
         IHAASCB
         EJECT
         IHAASXB
         END
```

Index

P

R

S

T

U

X

ITSO Technical Bulletin Evaluation RED000

RACF Version 2 Release 1
Installation and Implementation Guide

Publication No. GG24-4405-00

Your feedback is very important to help us maintain the quality of ITSO Bulletins. **Please fill out this questionnaire and return it using one of the following methods:**

- Mail it to the address on the back (postage paid in U.S. only)
- Give it to an IBM marketing representative for mailing
- Fax it to: Your International Access Code + 1 914 432 8246
- Send a note to REDBOOK@VNET.IBM.COM

Please rate on a scale of 1 to 5 the subjects below.
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction	____		
Organization of the book	____	Grammar/punctuation/spelling	____
Accuracy of the information	____	Ease of reading and understanding	____
Relevance of the information	____	Ease of finding information	____
Completeness of the information	____	Level of technical detail	____
Value of illustrations	____	Print quality	____

Please answer the following questions:

a) If you are an employee of IBM or its subsidiaries:

Do you provide billable services for 20% or more of your time? Yes____ No____

Are you in a Services Organization? Yes____ No____

b) Are you working in the USA? Yes____ No____

c) Was the Bulletin published in time for your needs? Yes____ No____

d) Did this Bulletin meet your needs? Yes____ No____

If no, please explain:

_____

_____

What other topics would you like to see in this Bulletin?

_____

_____

What other Technical Bulletins would you like to see published?

_____

Comments/Suggestions: (THANK YOU FOR YOUR FEEDBACK!)

_____ _____
Name Address

_____ _____
Company or Organization

_____ _____
Phone No.

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM International Technical Support Organization
Mail Station P099
522 SOUTH ROAD
POUGHKEEPSIE NY
USA 12601-5400

Fold and Tape **Please do not staple** Fold and Tape

Cut or Fold
Along Line

Printed in U.S.A.

GG24-4405-00