z/OS

SecureWay Security Server RACF Security Administrator's Guide

SA22-7683-01

z/OS

SecureWay Security Server RACF Security Administrator's Guide

SA22-7683-01

Note

Before using this information and the product it supports, be sure to read the general information under “Appendix G. Notices” on page 655.

Second Edition, October 2001

This is a major revision of SA22-7683-00. This edition applies to z/OS Version 1 Release 2 (5694–A01) and to subsequent releases and modifications until otherwise indicated in new editions.

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

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

International Business Machines Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie, NY 12601-5400
United States of America

FAX (United States & Canada): 1+845+432-9405
FAX (Other Countries):
Your International Access Code +1+845+432-9405

IBMLink (United States customers only): IBMUSM10(MHVRCFS)
Internet e-mail: mhvrcfs@us.ibm.com
World Wide Web: http://www.ibm.com/servers/eserver/zseries/zos/webqs.html

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:

- Title and order number of this book
- Page number or topic related to your comment

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

Contents

Tables

Figures

About This Book

This book contains information about Resource Access Control Facility (RACF), which is part of the SecureWay Security Server. The Security Server consists of these components:

- Resource Access Control Facility (RACF)
- DCE Security Server
- z/OS Firewall Technologies
- Lightweight Directory Access Protocol (LDAP) Server, which includes client and server function
- Open Cryptographic Enhanced Plug-ins (OCEP)
- Network Authentication Service

This book provides information to help the security administrator plan for and administer the RACF component of the SecureWay Security Server. For information about the other components of the Security Server, see the publications related to those components.

Who Should Use This Book

Security administrators, group administrators, and other administrators who are responsible for system data security and integrity on a z/OS system should use this book, for such tasks as:

- Planning how to use RACF to increase the security of the system
- Deciding which resources to protect
- Performing administration tasks
- Coordinating with administrators of other products

Readers should be familiar with RACF concepts and terminology. The readers of this manual should also be familiar with OS/390 or z/OS systems.

RACF overview information can be obtained from the RACF home page:

`http://www.ibm.com/servers/eserver/zseries/zos/racf/`

How to Use This Book

Much of this book describes how to protect resources, such as data sets, terminals, and others. In general, you first need to define users to RACF and set some RACF options. Then, depending on your security plan, you select classes of resources to protect and create resource profiles for them.

If you are reading this book for the first time, consider reading the following parts first:

- “Chapter 1. Introduction” on page 1
- “Chapter 2. Organizing for RACF Implementation” on page 35
- “Chapter 3. Defining Groups and Users” on page 49
- “Defining Profiles for General Resources” on page 196
- “Setting Up the Global Access Checking Table” on page 206
- “Getting Started with RACF (after First Installing RACF)” on page 305
- Appropriate portions of “Chapter 5. Specifying RACF Options” on page 111

Where to Find More Information

Where necessary, this book references information in other books. For complete titles and order numbers for all elements of z/OS, see *z/OS Information Roadmap*.

Softcopy Publications

The RACF library is available on the following CD-ROMs. The CD-ROM online library collections include the IBM Library Reader, which is a program that enables you to view the softcopy books.

SK3T-4269 *z/OS Version 1 Release 2 Collection*

This collection contains the set of unlicensed books for the current release of z/OS in both BookManager and Portable Document Format (PDF) files. You can view or print the PDF files with the Adobe Acrobat reader.

SK3T-4272 *z/OS SecureWay Security Server RACF Collection*

This softcopy collection kit contains the Security Server library for z/OS in both BookManager and Portable Document Format (PDF) files. You can view or print the PDF files with the Adobe Acrobat reader.

SK2T-2180 *Online Library OS/390 SecureWay Security Server RACF Information Package*

This softcopy collection kit contains the Security Server library for OS/390. It also contains the RACF/MVS Version 2 product libraries, the RACF/VM 1.10 product library, product books from the OS/390 and VM collections, International Technical Support Organization (ITSO) books (redbooks), and Washington System Center (WSC) books (orange books) that contain information related to RACF. The kit does not contain any licensed publications. By using this CD-ROM, you have access to RACF-related information from IBM products such as OS/390, VM/ESA, CICS, and NetView. For more information, see the advertisement at the back of the book.

SK2T-2177 *System Center Publication IBM S/390 Redbook Collection Collection*

This softcopy collection contains a set of redbooks that pertain to the S/390 platform and to host networking.

RACF Courses

The following RACF classroom courses are available:

ES840 *Implementing RACF Security for CICS/ESA and CICS/TS*

H3917 *Basics of OS/390 SecureWay Security Server RACF Administration*

H3927 *Effective RACF Administration*

H4020 *Exploiting the Features of OS/390 SecureWay Security Server RACF*

IBM provides a variety of educational offerings for RACF. For more information about classroom courses and other offerings, do any of the following:

- See your IBM representative
- Call 1-800-IBM-TEACH (1-800-426-8322)

Using LookAt to look up message explanations

LookAt is an online facility that allows you to look up explanations for z/OS messages and system abends.

Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

LookAt can be accessed from the Internet at:

`http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookat.html`

or from a TSO command line.

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

1. Go to http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookat.html.
2. Scroll to and click on the **News and Help** button.
3. Scroll to and click on the **Download LookAt from the Web** link.
4. Click on the ftp directory for the appropriate operating system and release.
5. Find the README file and follow its detailed instructions.

To find a message explanation from a TSO command line, simply enter: **lookat** *message-id* as in the following example:

```
lookat iec192i
```

This results in direct access to the message explanation for message IEC192I.

Note: Some messages have information in more than one book. For example, IEC192I has routing and descriptor codes listed in *z/OS MVS Routing and Descriptor Codes*. For such messages, LookAt prompts you to choose which book to open.

Accessing licensed books on the Web

z/OS licensed documentation in PDF format is available on the Internet at the IBM Resource Link Web site at:

`http://www.ibm.com/servers/resourcelink`

Licensed books are available only to customers with a z/OS license. Access to these books requires an IBM Resource Link Web userid and password, and a key code. With your z/OS order you received a memo that includes this key code.

To obtain your IBM Resource Link Web userid and password log on to:

`http://www.ibm.com/servers/resourcelink`

To register for access to the z/OS licensed books:

1. Log on to Resource Link using your Resource Link userid and password.
2. Click on **User Profiles** located on the left-hand navigation bar.
3. Click on **Access Profile.**
4. Click on **Request Access to Licensed books.**
5. Supply your key code where requested and click on the **Submit** button.

If you supplied the correct key code you will receive confirmation that your request is being processed. After your request is processed you will receive an e-mail confirmation.

Note: You cannot access the z/OS licensed books unless you have registered for access to them and received an e-mail confirmation informing you that your request has been processed.

To access the licensed books:

1. Log on to Resource Link using your Resource Link userid and password.
2. Click on **Library**.
3. Click on **zSeries**.
4. Click on **Software**.
5. Click on **z/OS**.
6. Access the licensed book by selecting the appropriate element.

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. A selected list of these books follows. 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.

G320-9279	*Systems Security Publications Bibliography*
GG22-9396	*Tutorial: Options for Tuning RACF*
GG24-3378	*DFSMS and RACF Usage Considerations*
GG24-3451	*Introduction to System and Network Security: Considerations, Options, and Techniques*
GG24-3524	*Network Security Involving the NetView Family of Products*
GG24-3970	*Elements of Security: RACF Overview - Student Notes*
GG24-3971	*Elements of Security: RACF Installation - Student Notes*
GG24-3972	*Elements of Security: RACF Advanced Topics - Student Notes*
GG24-3984	*RACF Macros and Exit Coding*
GG24-4282	*Secured Single Signon in a Client/Server Environment*
GG24-4453	*Enhanced Auditing Using the RACF SMF Data Unload Utility*
GG26-2005	*RACF Support for Open Systems Technical Presentation Guide*
GC28-1210	*System/390 MVS Sysplex Hardware and Software Migration*
SG24-4704	*OS/390 Security Services and RACF-DCE Interoperation*
SG24-4820	*OS/390 Security Server Audit Tool and Report Application*
SG24-5158	*Ready for e-business: OS/390 Security Server Enhancements*
SG24-5339	*The OS/390 Security Server Meets Tivoli: Managing RACF with Tivoli Security Products*

Other Sources of Information

IBM provides customer-accessible discussion areas where RACF may be discussed by customer and IBM participants. Other information is also available through the Internet.

IBM Discussion Areas

IBM provides the following discussion areas for RACF and security-related topics.

- **MVSRACF**

  MVSRACF is available to customers through IBM's TalkLink offering. To access MVSRACF from TalkLink:

  1. Select S390 (the S/390 Developers' Association).
  2. Use the fastpath keyword: MVSRACF.

- **SECURITY**

  SECURITY is available to customers through IBM's DialIBM offering, which may be known by other names in various countries. To access SECURITY:

  1. Use the CONFER fastpath option.
  2. Select the SECURITY CFORUM.

Contact your IBM representative for information on TalkLink, DialIBM, or equivalent offerings for your country and for more information on the availability of the MVSRACF and SECURITY discussions.

Internet Sources

The following resources are available through the Internet to provide additional information about the RACF library and other security-related topics:

- **Online Library**

  To view and print online versions of the z/OS publications, use this address:

  `http://www.ibm.com/servers/eserver/zseries/zos/bkserv/`

- **Redbooks**

  The redbooks that are produced by the International Technical Support Organization (ITSO) are available at the following address:

  `http://www.ibm.com/redbooks/`

- **Enterprise Systems Security**

  For more information about security on the S/390 platform, OS/390, and z/OS, including the elements that comprise the Security Server, use this address:

  `http://www.ibm.com/servers/eserver/zseries/zos/security/`

- **RACF Home Page**

  You can visit the RACF home page on the World Wide Web using this address:

  `http://www.ibm.com/servers/eserver/zseries/zos/racf/`

- **RACF-L Discussion List**

  Customers and IBM participants may also discuss RACF on the RACF-L discussion list. RACF-L is not operated or sponsored by IBM; it is run by the University of Georgia.

  To subscribe to the RACF-L discussion and receive postings, send a note to:

  `listserv@listserv.uga.edu`

  Include the following line in the body of the note, substituting your first name and last name as indicated:

  `subscribe racf-l` *`first_name last_name`*

  To post a question or response to RACF-L, send a note, including an appropriate `Subject:` line, to:

  `racf-l@listserv.uga.edu`

- **Sample Code**

You can get sample code, internally-developed tools, and exits to help you use RACF. This code works in our environment, at the time we make it available, but is not officially supported. Each tool or sample has a README file that describes the tool or sample and any restrictions on its use.

To access this code from a Web browser, go to the RACF home page and select the “Downloads” topic from the navigation bar, or go to ftp://ftp.software.ibm.com/eserver/zseries/zos/racf/.

The code is also available from `ftp.software.ibm.com` through anonymous FTP. To get access:

1. Log in as user **anonymous**.
2. Change the directory, as follows, to find the subdirectories that contain the sample code or tool you want to download:

   ```
   cd eserver/zseries/zos/racf/
   ```

An announcement will be posted on RACF-L, MVSRACF, and SECURITY CFORUM whenever something is added.

Note: Some Web browsers and some FTP clients (especially those using a graphical interface) might have problems using `ftp.software.ibm.com` because of inconsistencies in the way they implement the FTP protocols. If you have problems, you can try the following:

- Try to get access by using a Web browser and the links from the RACF home page.
- Use a different FTP client. If necessary, use a client that is based on command line interfaces instead of graphical interfaces.
- If your FTP client has configuration parameters for the type of remote system, configure it as UNIX instead of MVS.

Restrictions

Because the sample code and tools are not officially supported,

- There are no guaranteed enhancements.
- No APARs can be accepted.

To Request Copies of IBM Publications

Direct your request for copies of any IBM publication to your IBM representative or to the IBM branch office serving your locality.

There is also a toll-free customer support number (1-800-879-2755) available Monday through Friday from 6:30 a.m. through 5:00 p.m. Mountain Time. You can use this number to:

- Order or inquire about IBM publications
- Resolve any software manufacturing or delivery concerns
- Activate the program reorder form to provide faster and more convenient ordering of software updates

Summary of Changes

Summary of Changes
for SA22-7683–01
z/OS Version 1 Release 2

This book contains information previously presented in *z/OS SecureWay Security Server RACF Security Administrator's Guide*, SA22-7683-00, which supports z/OS Version 1 Release 1.

New Information

- "Defining Large Groups with the UNIVERSAL Attribute" on page 54
- "Limiting the Size of Your Access Lists" on page 205
- "Using RACFVARS with Mixed-Case Classes" on page 230
- "Using IRRDBU00 with Universal Groups" on page 323
- "Processing Universal Groups" on page 353
- "CREATE VIEW Privilege" on page 401
- "SETROPTS KERBLVL processing" on page 558
- "Java Archive (JAR) Privileges" on page 624

Changed Information

- "User Profiles" on page 62 includes new segments and new fields.
- "Reports Based on the Database Unload Utility (IRRDBU00)" on page 329 includes new reports supporting universal groups.
- Table 26 on page 337 includes new record types.
- "Using the RACF Remove ID Utility (IRRRID00)" on page 342 includes information on new general resource classes.
- "Chapter 11. Controlling Access to DB2 Objects" on page 387 includes information on new DB2 classes.
- "Chapter 20. RACF and SecureWay Network Authentication Service" on page 555 includes information on new key encryption options.
- "Appendix A. Description of RACF Classes" on page 565 includes new classes.
- "Appendix B. Summary of RACF Commands and Authorities" on page 573 includes new functions.
- "Appendix C. RACF Profile Summaries" on page 583 includes new options.
- "Appendix D. RACF/DB2 External Security Module: Authorization Checking" on page 595 includes information on new DB2 classes.

This book includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

You may notice changes in the style and structure of some content in this book—for example, headings that use uppercase for the first letter of initial words only, and procedures that have a different look and format. The changes are ongoing improvements to the consistency and retrievability of information in our books.

Chapter 1. Introduction

This chapter introduces you to using RACF to administer security on your system.

Over the past several years, it has become much easier to create and access computerized information. No longer is system access limited to a handful of highly skilled programmers; information can now be created and accessed by almost anyone who takes a little time to become familiar with the newer, easier-to-use, high-level inquiry languages. As a result of this improved ease of use, the number of people using computer systems has increased dramatically. More and more people are becoming increasingly dependent on computer systems and the information they store in these systems.

As the general computer literacy and the number of people using computers has increased, the need for data security has taken on a new level of importance. No longer can the installation depend on keeping data secure simply because no one knows how to access the data. Further, making data secure does not mean just making confidential information inaccessible to those who should not see it; it means preventing the inadvertent destruction of files by people who may not even know that they are improperly manipulating data.

As the security administrator, it is your job to ensure that your installation's data is properly protected. RACF can help you do this.

How RACF Meets Security Needs

The RACF licensed program satisfies the preferences of the end user without compromising any of the concerns raised by security personnel. The RACF approach to data security is to provide an access control mechanism that:

- Offers effective user verification, resource authorization, and logging capabilities
- Supports the concept of user accountability
- Is flexible
- Has little noticeable effect on the majority of end users, and little or no impact on an installation's current operation
- Is easy to install and maintain

User Identification and Verification

RACF controls access to and protects resources. For a software access control mechanism to work effectively, it must first *identify* the person who is trying to gain access to the system, and then *verify* that the user is really that person.

RACF uses a *user ID* and a system-encrypted *password* to perform its user identification and verification. When you define a user to RACF, you assign a user

ID and temporary password. The user ID identifies the person to the system as a RACF user. The password verifies the user's identity.

The temporary password permits initial entry to the system, at which time the person is required to choose a new password. Unless the user divulges it, no one else knows the user ID-password combination.

Note: During terminal processing, RACF allows the use of an operator identification card (OIDCARD) in place of, or in addition to, the password. (The OIDCARD information is also encrypted.) By requiring a user to know both the correct password and the correct OIDCARD, you have increased assurance that the proper user has entered the user ID.

The secured signon function provides an alternative to the RACF password called a PassTicket, which allows workstations and client machines to communicate with a host without using a RACF password. Using this function can enhance security across a network. For more information, see "Using the Secured Signon Function" on page 240.

Authorization Checking

Having identified a valid user, the software access control mechanism must next control interaction between the user and the system resources. It must authorize not only what resources that user may access, but also in what way the user may access them, such as for reading only, or for updating as well as reading. This controlled interaction, or authorization checking, is shown in Figure 1 on page 4. Before this activity can take place, however, someone with the proper authority at the installation must establish the constraints that govern those interactions.

With RACF, you are responsible for protecting the system resources (data sets, tape and DASD volumes, IMS and CICS transactions, TSO logon information, and terminals) and for issuing the authorities by which those resources are made available to users. RACF records your assignments in *profiles* stored in the RACF database. RACF then refers to the information in the profiles to decide if a user should be permitted to access a system resource.

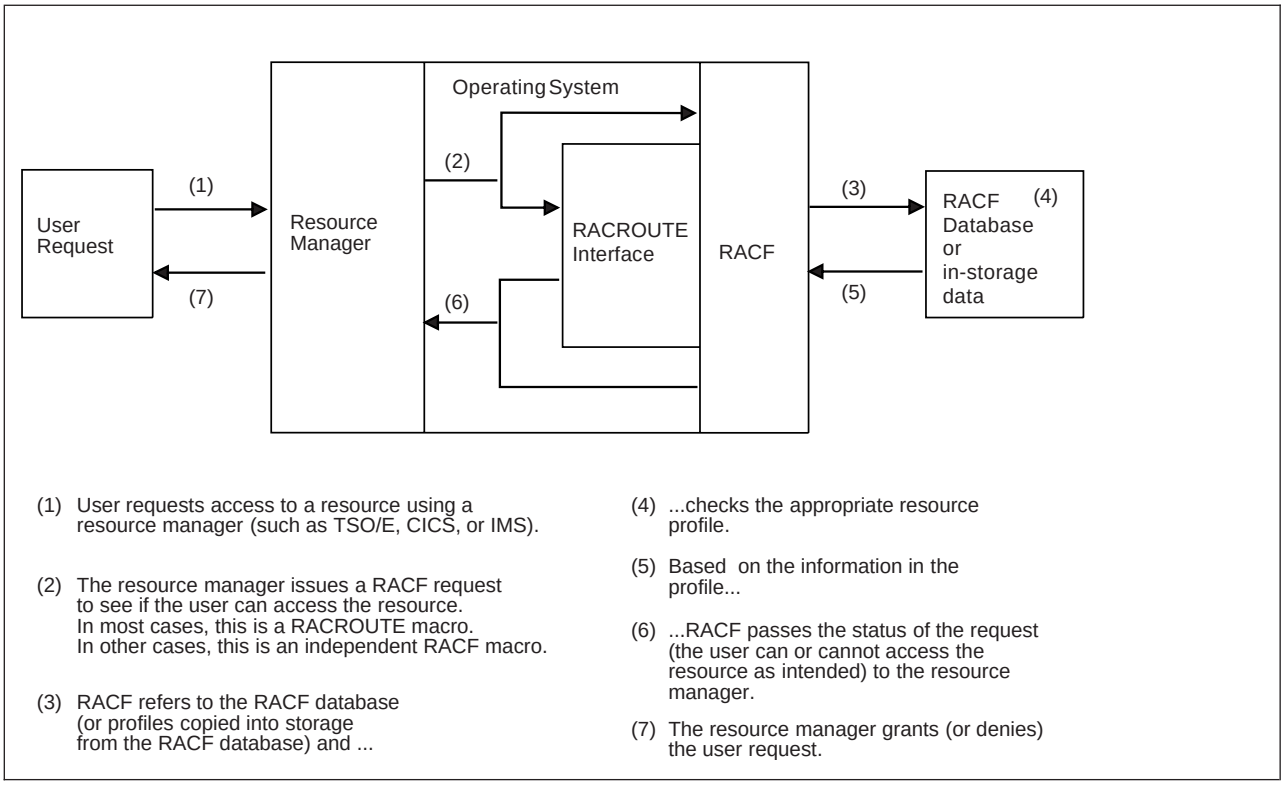


Figure 1. RACF Authorization Checking

Logging and Reporting

The ability to log information, such as attempted accesses to a resource, and to generate reports containing that information can prove useful to a resource owner, and is very important to a smoothly functioning security system.

Because RACF can identify and verify a user's user ID and recognize which resources the user can access, RACF can record the events where user-resource interaction has been attempted. This function records actual access activities or variances from the expected use of the system.

RACF has a number of logging and reporting functions that allow a resource owner to identify users who attempt to access the resource. In addition, you and your auditor can use these functions to log all detected successful and unsuccessful attempts to access the RACF database and RACF-protected resources. Logging all access attempts allows you to detect possible security exposures or threats. The logging and reporting functions are:

- **Logging:** RACF writes records to the system management facility (SMF) for detected, unauthorized attempts to enter the system. Optionally, RACF writes records to SMF for authorized attempts and detected, unauthorized attempts to:
  - Access RACF-protected resources
  - Issue RACF commands
  - Modify profiles on the RACF database

  RACF writes these records to an SMF data set. To list SMF records, you can use either the RACF SMF data unload utility (IRRADU00) or the RACF report writer.

- With the SMF data unload utility, you can translate the RACF SMF records into a format you can browse or upload to a database, query, or reporting package, such as DB2.
- With the report writer, you can select RACF SMF records to produce the reports. Because the RACF report writer was stabilized at the RACF 1.9.2 level, it cannot produce reports for all records beyond that release.

You should keep in mind that, for each logging activity that RACF performs, there is a corresponding increase in RACF and SMF processing.

For more information on logging and auditing, see *z/OS SecureWay Security Server RACF Auditor's Guide*. For information on how to specify logging and auditing functions, see *z/OS SecureWay Security Server RACF Command Language Reference*.

- **Sending Messages:** RACF sends messages to the security console for detected, unauthorized attempts to enter the system and for detected, unauthorized attempts to access RACF-protected resources or modify profiles on the RACF database.

  As well as sending resource access violation messages only to the security console, RACF allows you to send a message to a RACF-defined TSO user. Each resource profile can contain the name of a user to be notified when RACF denies access to the resource. If the user is not logged on to the system at the time of the violation, the user receives the message when logging on.

  If you are auditing access attempts, and you have selected the RACF function that issues a warning message instead of failing an invalid access attempt (to allow for a more orderly migration to a RACF-protected system), RACF records each attempted access. For each access attempt that would have failed, RACF sends a warning message (ICH408I) to the accessor, but allows the access. If a *notify user* is specified in the resource profile, RACF also sends a message to that user.
- **Keeping Statistical Information:** Optionally, RACF can keep selected statistical information, such as the date, time, and number of times that a user enters the system and the number of times a single user accesses a specific resource. This information can help the installation analyze and control its computer operations more effectively. In addition, to allow the installation to track and maintain control over its users and resources, RACF provides commands that enable the installation to list the contents of the profiles in the RACF database.

User Accountability

Individual accountability should probably be one of your installation's prime security objectives. A user who can be held individually accountable for actions is less likely to make mistakes or take other actions that might disrupt or compromise operations at your installation.

When an individual user accesses the system through a terminal, the concept of individual user identity is fairly obvious. With a group of production programs, however, it may be less clear just who the user is. (Is it the application owner, the job scheduling person, or the console operator?)

RACF offers you the ability to assign each user a unique identifier. (Of course, whether you establish this degree of accountability in all cases is an installation decision.)

In addition, RACF permits you to assign each user to one or more groups, which are simply collections of users having common access requirements.

RACF Users

A RACF user is identified by an alphanumeric *user ID* that RACF associates with the user. Note, however, that a RACF user need not be an individual. For example, a user ID can be associated with a started procedure. In addition, in many systems today a “user” is equated with a function, rather than an individual. For example, a service bureau customer may comprise several people who submit work as a single user. Their jobs are simply charged to a single account number. From the security standpoint, as mentioned before, equating a user ID with anything other than an individual can be undesirable because individual accountability is lost. However, it is up to the installation, through you, to decide how much individual accountability is required.

RACF Groups

A RACF group is normally a collection of users with common access requirements. As such, it is an administrative convenience, because it can simplify the maintenance of access lists in resource profiles. By adding a user to a group, you can give that user access to all of the resources that the group has access to. Likewise, by removing a user from a group, you can prevent the user from accessing those resources. You can also use groups as *holding* groups or *data control groups*. For more information, see “Defining RACF Groups” on page 50.

The group concept is very flexible; a RACF group can be equated with almost any logical entity, such as a project, department, application, service bureau customer, operations group, or systems group. Further, individual users can be connected to any number of groups. Membership and authority in these groups can be used to control the scope of a user’s activity.

What RACF Controls

You can use RACF to control access to:

- The system. You can require that all users, including TSO users, MVS system operators, and JES system operators (except operators on locally-attached JES3 consoles), log on and supply a password when logging on or submitting batch jobs. You can also require that all users supply a security label when logging on or submitting batch jobs.
- Many subsystems and applications, including:
  - JES, including job names, JES commands, consoles, JES input devices, spool, writers (printers) and others
  - TSO
  - IMS
  - CICS
  - Storage Management Subsystem (SMS)
  - DB2
  - VTAM
  - APPC sessions
- Terminals
- MVS and JES consoles
- Data, including:
  - Catalogs
  - DASD and tape data sets
  - DASD and tape volumes
  - SYSIN and SYSOUT data on the JES spool
  - Message traffic

- Load modules (programs) for execution only, or copying as well
- IMS/CICS transactions
- CICS files, journals, and so forth
- Installation-defined resources
- APPC transactions and other resources

For more information, see “Protecting General Resources” on page 21.

How Users and Groups Are Authorized to Access Resources

Basically, a user’s authority to access a resource while operating in a RACF-protected system at any time is determined by a combination of these factors:

- The user’s *identity*
- The user’s *attributes*
- The user’s *group authorities*
- The *security classification* of the user and the resource profile
- The *access authority* specified in the resource profile

Identity: When defining a user, the security administrator assigns a user ID consisting of 1–8 characters. This is the user ID with which the user logs on to the system (or submits a batch job). When a user attempts to access RACF-protected resources, RACF uses the user ID to determine the user’s access to those resources.

Attributes: The security administrator or a delegate can assign attributes to each RACF-defined user. The attributes determine various extraordinary privileges and limitations a user has when using the system. Attributes are classified as either user-level attributes (or, simply, user attributes) or group-level attributes:

- User attributes: You can assign the SPECIAL, AUDITOR, OPERATIONS, CLAUTH, GRPACC, ADSP, and REVOKE attributes at the system level. When you assign attributes at the system level, the privileges and limitations apply across the entire system. For detailed information about these attributes, see “Assigning Optional User Attributes” on page 17 and “User Attributes” on page 73.
- Group-level attributes: When you assign an attribute at the group level, RACF confines the privileges or limitations conveyed by the attribute to the group to which it applies (and to resources, users, and groups that fall within the scope of that group). For more information about the group-SPECIAL, group-AUDITOR, and group-OPERATIONS attributes, see “Assigning Optional User Attributes” on page 17 and “User Attributes” on page 73.

Group Authorities: Each user that you define must be assigned (connected) to at least one group (called the user’s default group). The security administrator or group administrator can assign a specific level of “group authority” to each user of a group. The group authorities are USE, CREATE, CONNECT, and JOIN.

If a user has USE group authority within a group, the user can access resources to which the group is authorized.

CREATE, CONNECT, and JOIN also enable the user to access resources to which the group is authorized. However, these group authorities also give the user administrative responsibilities and privileges. The USE, CREATE, CONNECT, and JOIN group authorities are described in detail in “Defining RACF Groups” on page 50.

Note: If a user is added to a RACF group (via the CONNECT command) after that user has already logged on, that user will have to log off and log back on to have authority based on that group when accessing resources in classes that have been RACLISTed.

If a user is deleted from a RACF group (via the REMOVE command) after that user is already logged on, that user will have to log off and log back on to not have authority based on that group when accessing resources in classes that have been RACLISTed.

Security Classification: Each user and each resource can have a security classification specified in its profile. The security classification can be a security level, one or more security categories, or both. A security *label* is an installation-defined name that refers to a combination of a security level and zero or more security categories. A security *level* is an installation-defined name that corresponds to a numerical security level (the higher the number, the higher the security level). A security *category* is an installation-defined name corresponding to a department or an area within an organization that has similar security requirements.

When a user requests access to a resource that has a security classification, RACF compares the security classification of the user with the security classification of the resource. For more information on security classifications, see “Chapter 4. Classifying Users and Data” on page 95.

Access Authority: The access authority determines to what extent the specified user or group can use the resource. The owner of a profile protecting a data set or general resource (such as a tape volume or terminal) can grant or deny a user or group access to that resource by including the user ID or group name in the resource profile’s access list. Associated with each user ID or group name is an access authority that determines whether the user or group can access the resource, and if they can access the resource, how they can use it.

The access authorities are NONE, EXECUTE, READ, UPDATE, CONTROL, and ALTER (see Table 47 on page 580).

For data set profiles, an entry in the access list may also contain the name of a program that is associated with the user ID and the access authority. In this case, the user must be executing that program to access the resource. For more information, see “Program Access to Data Sets (PADS)” on page 252.

For general resource profiles, an entry in the access list may also contain the name of a RACF-defined terminal or a console, JES input device, or partner LU that is associated with the user ID and the access authority. In this case, the user must be using the terminal, the console, the JES input device, or a partner LU name to access the resource. For more information, see “Conditional Access Lists for General Resource Profiles” on page 206.

Each resource also has a universal access authority (UACC) associated with it. The UACC can be NONE, EXECUTE, READ, UPDATE, CONTROL, or ALTER. The UACC is the access authority allowed to any group or non-restricted user who is not authorized in the access list. UACC applies to all users, whether or not they are RACF-defined, unless they are defined with the RESTRICTED attribute. For information about assigning the RESTRICTED attribute, see “Defining Restricted User IDs” on page 87.

Using ID(*) on the Access List: If you have some users who are not defined to RACF, you can use the `ID(*)` entry on the access list instead of UACC to ensure that only RACF-defined users, except those with the RESTRICTED attribute, can access the resource. The following examples illustrate the difference between UACC(READ) and ID(*) ACCESS(READ):

- To allow *all* users on the system to use a terminal, specify UACC(READ) for the profile, as follows:

  ```
  RDEFINE TERMINAL profile-name UACC(READ)
  ```

- To allow only *RACF-defined* users on the system to use a terminal, specify UACC(NONE) for the profile, then issue the PERMIT command with ID(*) and ACCESS(READ) specified:

  ```
  RDEFINE TERMINAL profile-name UACC(NONE)
  PERMIT profile-name CLASS(TERMINAL) ID(*) ACCESS(READ)
  ```

  Note: Neither the `ID(*)` entry on the access list nor the UACC is used to allow a restricted user to access a RACF-protected resource.

RACF Profiles

As the security administrator or a delegate defines authorized users, groups, and protected resources, RACF builds *profiles,* which contain the information RACF uses to control access to the protected resources. Each profile is owned by a user or group. (By default, the owner of a profile is the user who creates it.)

You can work with the following types of profiles:

- User profiles
- Group profiles
- Data set profiles
- General resource profiles

User and group profiles contain descriptions of the authorized users of a RACF-protected system. Data set and general resource profiles contain descriptions of the resources and the levels of authority that are necessary to access these resources.

Flexibility

Because the security requirements at every installation differ, RACF is flexible enough to assist each installation in meeting its own security objectives. There are a number of ways RACF accomplishes this:

- **Administrative Control:** RACF allows you a wide range of choices in controlling access to your installation's resources. RACF allows you to use either centralized or decentralized administration techniques by permitting you to delegate authority, establish appropriate group ownership structures, and specify various group-related user attributes. In addition, RACF provides a wide range of processing options and installation exits.

  All RACF command functions, except those performed by the RVARY command, the RACF report writer command (RACFRW), and the block update command (BLKUPD), have Interactive System Productivity Facility (ISPF) entry panels and associated help panels. These panels make it easy to enter command options on TSO.

- **Generic Profiles:** RACF generic profiles allow you, your group administrators, and other users to define profiles that consolidate the security requirements of several similarly-named resources that have the same access requirements.

- **Protection of Installation-Defined Resources:** RACF allows you to protect your own installation-defined resource classes. To do this, add an entry in the class

descriptor table (CDT) for the class of the resource, create profiles in the class, and, when a user requests access to a resource (or takes an action you wish to control), issue the RACROUTE REQUEST=AUTH macro from your application. You can control which users and groups can access each resource in the class by defining profiles in the class. The profiles can include access lists and other information such as auditing, security labels, and so forth, as with profiles in classes supplied by IBM. (See *z/OS SecureWay Security Server RACF System Programmer's Guide* for more information on creating installation-defined resource classes.)

- **Installation Exits:** RACF installation exits allow you to tailor RACF to specific needs of your installation. For more information, see "Using RACF Installation Exits to Customize RACF" on page 24.

Because of RACF's flexible design, you and your technical support personnel can tailor RACF to operate smoothly within the local operating environment.

Implementing Government Security Levels

If your installation's computing system must meet security criteria that have been established by the United States Department of Defense, RACF can help. For more information, see "Government Security Levels".

RACF Transparency

No users want their data destroyed or altered by other individuals (or themselves) except when they specifically intend it. Unfortunately, users of all types are often reluctant to take steps to protect what they have created. It is not uncommon to see live data used as test data, or to see data deliberately underclassified to avoid having to use the security procedures that the appropriate classification would demand. In many cases, people find it easier to ignore security procedures than to use them. Even conscientious users can forget to protect a critical piece of data. The solution to implementing effective security measures, then, is to provide a security system that is transparent (painless) to the user.

With RACF, end users need not be aware that their data is being protected for them. Security and group administrators can use generic profiles to make using RACF transparent to the majority of the installation's end users. Administrators can also use profile modelling to enhance RACF's transparency.

Government Security Levels

The United States Department of Defense (DoD) has established security criteria for its computer systems and for those systems that perform government work under contract. Each system is evaluated and awarded a security rating, depending on the extent the system protects resources and its own processing. These ratings are, in order of least to most secure, D, C1, C2, B1, B2, B3, and A1.

C2 Security Level

DoD describes a computing system that operates at the C2 security level as one that enforces discretionary access control by making users individually accountable for their actions through the use of logon procedures, auditing of security-related events, and resource isolation. Such a system is called a **C2 trusted computing base.**

C2 Criteria

DoD has established specific criteria in the following areas for operating a system at the C2 security level:

- Discretionary access control (DAC)
- Auditability of security-related events
- Resource reuse
- Identification and authentication

Discretionary Access Control (DAC): Discretionary access control is a method of limiting access to resources (such as data sets) based on the identity of users or groups to which the users belong. DAC protects all system resources from unauthorized access down to a single user. A user who does not have permission to access a resource can only be granted this permission by the resource's owner.

Auditability of Security-Related Events: Auditability of security-related events is the recording of facts that describe a security-related event in a computing system. These facts include the time and date of the event, the name of the event, the name of the system resources affected by the event, the name of the user who invoked the event, and so forth.

Resource Reuse: Resource reuse is a practice that ensures all system resources (such as tape data sets) that are reused, reassigned, or reallocated are purged of all residual data, including encrypted data, belonging to the former owner.

Identification and Authentication: Identification and authentication is a method of enforcing individual accountability by providing a way for each user to be uniquely identified. Users must then have their identity associated with any security-related, audited action they might take.

C2 Configuration Requirements

The U.S. Federal government previously evaluated for C2 certification. RACF was part of the C2 trusted computing base for the system.

B1 Security Level

The requirements that the DoD sets forth concern two major areas: security policy and accountability. A security policy is a set of rules or practices that regulate how an organization handles its sensitive data. Accountability refers to the ability to establish a relationship between an action and the user responsible for the action.

B1 Security Criteria

The security policy that you implement in a B1 environment has as its major requirement a system of access controls that not only prevents individuals from accessing information at a classification for which they are not authorized, but also prevents individuals from declassifying information. The system must protect resources of different levels of sensitivity.

There are specific requirements in each of the following areas:

- Mandatory access control (MAC)
- Discretionary access control (DAC)
- Resource reuse
- Security Labels
- Identification and authentication
- Auditing
- The trusted computing base

Mandatory Access Control (MAC): Mandatory access control is a method of limiting access to resources based on the sensitivity of the information that the resource contains and the authorization of the user to access information with that level of sensitivity.

You define the sensitivity of the resource by means of a label. This label indicates the level or classification of the information (for example, Restricted, Confidential, or Internal). Within that level, you define the category to which the information belongs (such as Project A or Project B). Users can access only the information in a resource to which their security labels entitle them. If the user's security label does not have enough authority, the user cannot access the information in the resource.

Discretionary Access Control (DAC): Discretionary access control is the same for B1 as it is for C2.

Resource Reuse: Resource reuse is the same for B1 as it is for C2.

Security Labels: Security labels are associated with all users and resources in the system. The system uses these labels to determine if access to a resource is allowed under the mandatory access control (MAC) rules. Security labels, maintained in the RACF database, are usually defined by the security administrator and can be changed only by that person.

When a resource is exported to a device attached to a system, the security label of the resource remains in effect. Whether the resource resides on a single-level device, such as a tape drive that does not process information at different levels of security concurrently, or a multilevel device, which is able to process data at different security levels concurrently, the system continues to associate the security label with the resource.

The system provides security labels on each page of print output as a default. The system allows a user to request that no security labels be printed; however, the system is able to audit all such requests.

Identification and Authentication: Identification and authentication is the same for B1 as it is for C2.

Auditing: Auditing is the same for B1 as it is for C2 with the following additions:
- An audit record contains the audited resource's security label.
- More selective options are available for audit reports.
- The system can audit any override of labeling on printed output.

The Trusted Computing Base: The "trusted computing base" is a "trusted computer system" that uses both hardware and software to ensure that these criteria are met. The security-relevant portion of the system, called the "trusted computing base," is made up of many separate hardware and software components. It is the combination of these components that enforces the security requirements of a system.

B1 Configuration Requirements

See *MVS/ESA SP V5 Planning: B1 Security (GC28–1440)* for information about operating at a B1 security level.

RACF meets the B1 level of trust, with the following exceptions:

- RACF support for z/OS UNIX System Services (z/OS UNIX) meets the C2 level of trust.

- When secured signon authentication is used from a remote node to authenticate to the MVS host, the host system does not meet B1 or C2 criteria.

The lists in this section have been included here as a reminder of what must be installed when certain options are to be used, some of which are SETROPTS MLS, MLACTIVE, MLQUIET and MLSTABLE.

Administering Security

The security administrator's job can range from helping high-level management initially define corporate security policy to authorizing individual end users to access RACF-protected resources. As security administrator, you are responsible for implementing RACF at your installation. You have the authority to review and approve all implementation phases, select the resources to be protected, and plan the order in which protection is implemented. You are the authority for all RACF implementation questions. You decide the degree to which decentralization of security controls takes place. You create profiles for the implementation team, select the team members, and direct their efforts.

Delegating Administration Tasks

Although you have responsibility for overall security at your installation, you can decentralize much of the security operation by delegating various RACF security responsibilities to assistants. You can appoint:

- **Group Administrators:** Group administrators have many of the duties and responsibilities of a security administrator, but at a less inclusive level. Typically, a group administrator is responsible for defining the access requirements for the resources belonging to a single group. In some cases, the group administrator may delegate responsibilities in the same way as you delegated yours.
- **Technical Support:** The technical support person is typically a system programmer whose job is to install operating systems, apply fixes to problems in the operating systems, and write necessary programs to interface between operating system programs and application programs. The technical support person is responsible for providing you with technical assistance, installing and maintaining RACF, and extending RACF to meet installation needs, as you direct. Technical support activities can include maintaining the RACF database.
- **Auditor:** The auditor supports the security implementation by ensuring that the levels of protection are adequate and that security exposures are reduced or eliminated. In addition, the auditor monitors operations to ensure that security procedures are being carried out properly.

In certain installations, it is possible that some of these functions might be combined. Further, the amount of delegation varies from installation to installation. In some installations, there may be much delegation of authority, and there may be more than one technical support person or more than two levels of group administrators. Similarly, other roles may differ somewhat from the way they are described in this publication.

For details about defining profiles to delegate administration tasks, see "Planning for Profiles in the FACILITY Class" on page 218.

Administering Security When a VM System Shares the RACF Database

The Security Server can be installed and run only on z/OS and OS/390 systems. However, your installation can share the RACF database with a VM system on which RACF is running. A RACF database that is shared with a VM system can

contain information about users and resources that is relevant only to that VM system. Although you can perform some RACF administration tasks for your VM system by using commands you issue on z/OS or OS/390, this library does not try to describe those tasks. For complete information about administering RACF on VM, see the RACF publications that apply to the RACF release you are running.

If your installation is sharing the RACF database with a VM system running RACF Version 1.10.0 or later, administration for OpenExtensions VM users and groups can be performed from your z/OS or OS/390 systems. Note that changing OpenExtensions VM user identifiers (UIDs) and group identifiers (GIDs) creates corresponding updates in the VMPOSIX class profiles.

Using RACF Commands or Panels

After you have planned for RACF implementation (see “Chapter 2. Organizing for RACF Implementation” on page 35), you can perform security and group administration tasks by using various RACF commands. For example, you can use the ADDGROUP command to define a new group as a subgroup of an existing group; you can use the ADDUSER command to define a new user and connect the user to the user’s default group; you can use the ADDSD command to protect a DASD data set, and so on. (Sample command sequences are given throughout this book for administrative tasks. See *z/OS SecureWay Security Server RACF Command Language Reference* for the attributes and authorities you need to use RACF commands.)

The RACF commands include operands with which you specify the various user attributes, group authorities, and access authorities. RACF places the information it receives from the commands into various profiles (user, group, data set, and general resource profiles), which it keeps in the RACF database and uses to control subsequent access to resources.

As an alternative to using RACF commands to perform administration tasks, you can use RACF’s ISPF panels if the ISPF product is installed at your location. If you use the panels, you don’t need to memorize command or operand names; you only need to complete the appropriate information on the proper panels.

Choosing between Using RACF TSO Commands and ISPF Panels

In general, you can perform the same RACF functions using RACF TSO commands and ISPF panels.

The **RACF TSO commands** provide the following advantages:

- Entering commands can be faster than displaying many panels in sequence.
- Using commands from book descriptions should be relatively straightforward. The examples in the books are generally command examples.
- Getting online help for RACF TSO commands

  You can get online help for the RACF TSO commands documented in *z/OS SecureWay Security Server RACF Command Language Reference*.
  - To see online help for the PERMIT command, for example, enter:

    ```
    HELP PERMIT
    ```
  - To limit the information displayed, specify operands on the HELP command. For example, to see only the syntax of the PERMIT command, enter:

    ```
    HELP PERMIT SYNTAX
    ```

Note: TSO online help is not available when RACF commands are entered as RACF operator commands.

- Getting message ID information

  If a RACF TSO command fails, you will receive a message. If you do not get a message ID, enter:

  ```
  PROFILE MSGID
  ```

  Reenter the RACF TSO command that failed. The message appears with the message ID. See the *z/OS SecureWay Security Server RACF Messages and Codes* for help if the message ID starts with ICH or IRR.

  Note: PROFILE MSGID cannot be entered as a RACF operator command.

The **ISPF panels** provide the following advantages:

- When you use the panels, you avoid having to memorize a command and type it correctly. Panels can be especially useful if the command is complex or you perform a task infrequently.
- ISPF creates in the ISPF log a summary record of the work that you do. Unless you use the TSO session manager, the RACF commands do not create such a record.
- From the panels, you can press the HELP key to display brief descriptions of the fields on the panels.
- The options chosen when installing the RACF panels determine whether output (for example, profile listings, search results, and RACF options) is displayed in a scrollable form.
- The ISPF panels for working with password rules allow you to enter all of the password rules on one panel. Figure 2 shows one of these panels.

```
                    RACF - SET PASSWORD FORMAT RULES
COMMAND ===>
               _
Enter PASSWORD FORMAT RULES:
                       MINIMUM  MAXIMUM
                       LENGTH   LENGTH     FORMAT
             RULE 1:   __       __         ________
             RULE 2:   __       __         ________
             RULE 3:   __       __         ________
             RULE 4:   __       __         ________
             RULE 5:   __       __         ________
             RULE 6:   __       __         ________
             RULE 7:   __       __         ________
             RULE 8:   __       __         ________
To cancel an existing rule, enter NO for MINIMUM LENGTH.
To specify FORMAT, use the following codes for each character position:
  * = Any character    A = Alphabetic    C = Consonant     V = Vowel
  W = No Vowel         N = Numeric       L = Alphanumeric
```

Figure 2. Sample ISPF Panel for RACF

RACF Group and User Structure

Two of the fundamental elements of RACF are users and groups. Users, of course, are the many people who log on to a system, each with a unique user ID. Administration of a small number of users is not too difficult. However, when there are thousands of users, administration becomes a very large task. To make this task more manageable, the concept of groups was developed.

Introduction

A group is a RACF entity with which any number of users are associated. Usually, the users in a group have some logical relationship to one another. The relationship used most frequently is members of a department. Many installations pattern their group-user structure after their organization charts.

At the top of the RACF group-user structure is a group called SYS1. When you install RACF, it defines this group for you. The SYS1 group is the highest group in the total RACF group-user structure. You can define your system administrator and system auditor as members of this group. The system administrator has the SPECIAL attribute and the system auditor has the AUDITOR attribute. The significance of SPECIAL and group-SPECIAL and AUDITOR and group-AUDITOR, and the differences between them, are described in later sections.

Defining Users and Groups

You define users to RACF by issuing RACF commands that include various user attributes, as well as other control information that RACF uses. The following are some of the commands you might use in your user-definition tasks. For a more complete description of the process of defining users, see “Defining Users” on page 61. For complete descriptions of RACF commands, see the *z/OS SecureWay Security Server RACF Command Language Reference*.

Commands for User Administration

ADDUSER Add a user profile to RACF.

ALTUSER Change a user’s RACF profile.

CONNECT Connect a user to a group.

DELUSER Delete a user profile from RACF and remove connection to all groups.

REMOVE Remove a user from a group and assign a new owner for group data sets owned by the removed user.

LISTUSER Display the contents of a user’s profile.

PERMIT Permit a user to access a resource (or deny access to a resource).

PASSWORD Change a user’s password.

In addition to defining individual users, you can define groups of users. Group members can share common access authorities to a protected resource.

One benefit of grouping users is that you can authorize the entire group, as a single unit, to access a protected resource. Another benefit is that attributes such as OPERATIONS can be assigned so that a given user has that attribute only when connected to a specific group, and the attribute is only effective for resources within the scope of that group.

The following are some of the commands you might use in your group-definition tasks.

Commands for Group Administration

ADDGROUP Define a new group (a subgroup of an existing group).

ALTGROUP Assign a subgroup to a new superior group.

DELGROUP Delete one or more groups.

LISTGRP Display the contents of a group profile.

CONNECT Connect a user to a group.

REMOVE Remove a user from a group and assign a new owner for group data sets owned by the removed user.

PERMIT Permit a group of users to access a resource (or deny them access to a resource).

Assigning Optional User Attributes

You can assign user attributes by specifying operands on RACF commands. User attributes describe various extraordinary privileges, limitations, and processing environments that can be assigned to specified users in a RACF-protected system.

You can assign user attributes at either the system level or at the group level. When assigned at the system level, attributes are effective for the entire RACF-protected system. When assigned at the group level, their effect is limited to profiles that are within the *scope of the group*.

The scope of control of a group-level attribute percolates down through a group-ownership structure from group to subgroup to subgroup, and so on. Percolation is halted (and therefore the scope of control of the group-level attribute is ended) when a subgroup is owned by a user instead of a superior group. Figure 3 shows an example of the scope of control of an attribute assigned at the group level.

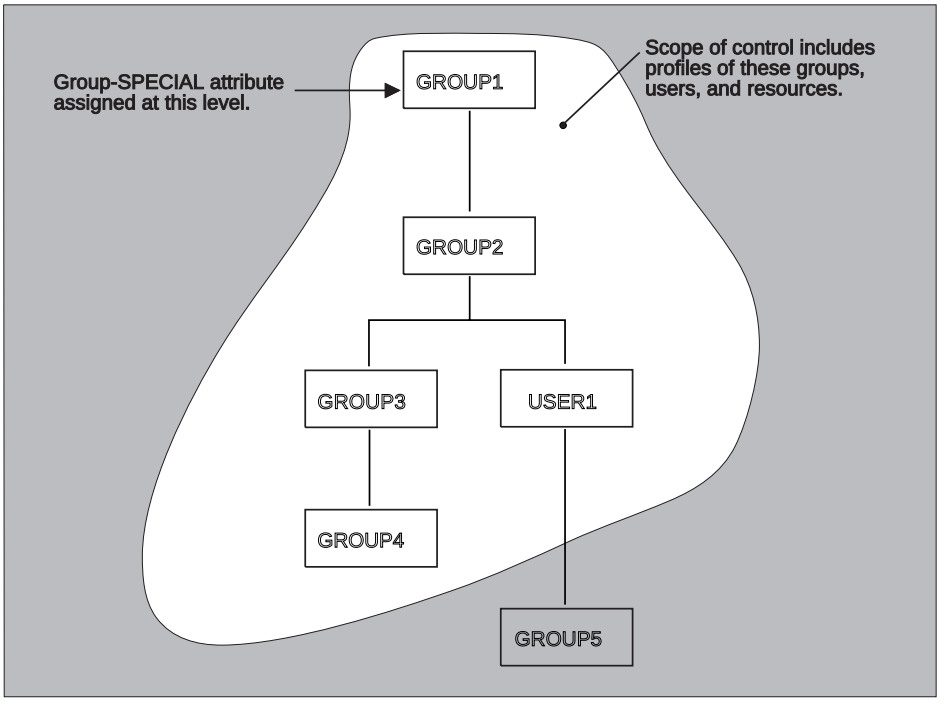


Figure 3. Scope of Control of an Attribute Assigned at the Group Level

Figure 3 shows a group ownership structure. In this figure, GROUP1 owns GROUP2, GROUP2 owns GROUP3 and USER1, and so on. A user who is connected to GROUP1 with the group-SPECIAL attribute has an explicit scope of control as shown in the figure. That is, the user cannot modify any profiles owned

by GROUP5. Table 1 lists and describes attributes that can be assigned at the user and group level. For a more complete description, see “Chapter 3. Defining Groups and Users” on page 49.

Table 1. User Attributes

User Attribute	Description
SPECIAL	When you assign it at the system level, the SPECIAL attribute gives the user full control over all of the RACF profiles in the RACF database. At the system level, the SPECIAL attribute allows the user to issue all RACF commands. When you assign the SPECIAL attribute at the group level, the *group-SPECIAL* user has full control over all of the resources that are within the scope of the group but cannot issue RACF commands that would have a global effect on RACF processing.
AUDITOR	When you assign it at the system level, the AUDITOR attribute gives the user full responsibility for auditing the security controls and use of system resources across the entire system. With the AUDITOR attribute at the system level, the user can specify logging options on the RACF commands and list the auditing options of any profiles using the RACF commands. In addition, the user can control additional logging to SMF for detecting changes and attempts to change the RACF database and for detecting accesses and attempts to access RACF-protected resources. When you assign the AUDITOR attribute at the group level (that is, when you assign the *group-AUDITOR* attribute), auditing authority is limited to resources that are within the scope of the group.
OPERATIONS	When you assign the OPERATIONS attribute at the system level, the user can perform any maintenance operations on RACF-protected resources, such as copying, reorganizing, cataloging, and scratching data. At the *group-OPERATIONS* level, authority to perform these operations is limited to resources that are within the scope of the group.
CLAUTH	The CLAUTH (class authority) attribute allows the user to define profiles in a specific RACF class. A user can have class authority for the USER class and any of the classes that are defined in the class descriptor table (CDT). Examples of classes that IBM supplies in the CDT are the TERMINAL class (for terminals) and the TAPEVOL class (for tape volumes). For a list of valid class names, see “Appendix A. Description of RACF Classes” on page 565. If the SETROPTS GENERICOWNER option is in effect, this authority is limited. See “Restricting the Creation of General Resource Profiles (GENERICOWNER Option)” on page 117.
GRPACC	When a user with the GRPACC attribute creates a data set profile for a group data set, RACF gives UPDATE access authority to other users in the group (if the user defining the profile is a member of that group). A group data set is a data set whose high-level qualifier, or the qualifier derived from the RACF naming convention table, is a RACF-defined group name.
ADSP	The ADSP attribute establishes an environment in which all permanent DASD data sets created by this user are automatically defined to RACF and protected with a discrete profile. ADSP can be assigned at the group level, in which case it is effective only when the user is connected to that group.
REVOKE	The REVOKE attribute prevents the RACF-defined user from entering the system. REVOKE can be assigned at the group level, in which case the user cannot enter the system and connect to that group.
RESTRICTED	The RESTRICTED attribute prevents a user from gaining access to a protected resource unless the user is specifically authorized on the access list. Global access checking, the `ID(*)` entry on the access list, and the UACC will not be used to allow a restricted user to access a protected resource.

Note: You and your delegates should assign the SPECIAL, AUDITOR, and OPERATIONS attributes to the minimum number of people necessary to administer security at the installation.

Assigning Group Authorities

Each user in a group may have different responsibilities for the group. These responsibilities may include creating resource profiles to be used by the group and adding new members to the group. You should assign a specific level of group authority to the user that is based on the user's responsibilities for administering and maintaining the group to which the user is connected. You can do this with the ADDUSER, ALTUSER, or CONNECT command.

The group authorities you can assign to a user are (in order of least to most authority): USE, CREATE, CONNECT, and JOIN. Each higher-level authority includes the lower levels of authority. The group authorities are defined generally as follows:

- The USE authority permits the user to access resources to which the group is authorized.
- The CREATE authority permits the user to create group data set profiles.
- The CONNECT authority enables the user to add RACF-defined users to the group.
- The JOIN authority enables the user to define new users and new groups.

For the specific details of each group authority, see "Group Authorities" on page 57.

Profiles Associated with Users and Groups

When you use RACF commands to define users and groups, the information RACF gathers from these commands is stored in profiles and placed in the RACF database. A general description of user and group profiles follows:

The User Profile: The user profile defines an individual user. Some of the things the user profile can contain are:

- Information about the user's identity, such as name and password (this can be masked, or encoded using RACF's implementation of the DES algorithm)
- System-wide user attributes
- The name of the user's default group
- Whether the user's security-related activities should be logged
- How often the user's password is to be changed
- The user's security categories, security level, and default security label
- The name of an optional model profile RACF uses when a user creates new data set profiles
- A TSO segment, which contains TSO logon information
- A DFP segment, which contains default DFP information for the user
- An OPERPARM segment, which contains initial information used when the user enters an extended MCS console session
- A CICS segment, which contains initial information used when the user signs on to CICS
- For z/OS UNIX, an OMVS segment, which contains z/OS UNIX information about the user
- For OpenExtensions VM, an OVM segment, which contains OpenExtensions VM information about the user

The Group Profile: The group profile defines a group. Some of the things the group profile can contain are:

- Information about the group, such as who owns it and what subgroups it has
- Whether the group is a universal group

- For non-universal groups, a list of all connected users (members)

  Note: Member lists for universal groups do not contain information about all connected users, only those users with group authorities higher than USE, and those with the group-SPECIAL, group-OPERATIONS, or group-AUDITOR attributes. For more information, see “Defining Large Groups with the UNIVERSAL Attribute” on page 54.
- For non-universal groups, the group authorities of each member

  Note: Information about members with group authority of USE is not available for universal groups.
- The name of an optional model profile RACF uses when a user creates new group data set profiles
- A DFP segment, which contains default DFP information for the group
- For z/OS UNIX, an OMVS segment, which contains z/OS UNIX information about the group
- For OpenExtensions VM, an OVM segment, which contains OpenExtensions VM information about the group

Protecting Resources

In the early releases of RACF, the only resources that were protected were data sets. Over the years, enhancements to RACF, applications have broadened the meaning of the term *resource* to include the following:

- Places in the system where data resides (such as data sets)
- Places in the system through which data passes during processing (such as terminals)
- The functions by which users work with data (such as commands)

Using RACF, you can protect resources so that only authorized users can access the resource in approved ways.

In general, you control access to a protected resource by creating a *discrete* or *generic* profile.

Discrete profiles protect only one resource. The name of the profile identifies to RACF which resource is protected. For example, a profile called ‘SMITH.REXX.EXEC’ in class DATASET would protect the data set named ‘SMITH.REXX.EXEC’.

Generic profiles protect one or more resources that have the same security requirements. In many cases, some of the characters in the resource names are the same. For example, a profile called ‘SMITH.**’ in class DATASET would protect all of SMITH’s data sets that did not have a more specific profile defined.

In most general resource classes, you can also provide a “top” generic profile that protects all of the resources that are not otherwise protected.

Note: A top generic profile for a class should have a profile name of ** (rather than *) so that you can issue the RLIST command to display the profile itself.

Using generic profiles can greatly reduce the amount of RACF profile maintenance done by a RACF administrator.

Examples of discrete and generic profiles are shown throughout this book.

Protecting Data Sets

RACF can protect the following kinds of data sets:

- VSAM data sets
- Data sets managed by the Storage Management Subsystem (SMS)
- Cataloged and uncataloged non-VSAM DASD data sets
- Tape data sets with standard labels
- Data sets that have the same name but reside on different volumes
- Generation data group (GDG) data sets

RACF protects data sets whether or not they are protected by passwords. When both RACF protection and password protection are applied to a data set, access to the data set is determined only through RACF authorization checking. That is, password protection is bypassed.

RACF protection has an advantage over password protection. With RACF protection, only authorized users can access the data set, With password protection, any user who knows the password can access the data set. Also, users can run jobs more easily using RACF protection because the system operator is not prompted for data set passwords for RACF-protected data sets that are accessed during a job.

To protect either a DASD or tape data set, a user issues the ADDSD command, which creates a data set profile and stores it in the RACF database. Alternatively, the user can specify the PROTECT=YES operand in the JCL or the PROTECT operand on the TSO ALLOCATE command. For tape data sets, the user can also predefine the tape volume using the RDEFINE command. (When protecting a tape data set, RACF also creates, under certain circumstances, a profile for the tape volume that contains the tape data set.)

You can protect data sets with either discrete or generic profiles. If a data set has unique access-authorization or logging requirements, you should define a discrete profile for it. If the requirements are the same for several data sets that share a common name structure, you can define a generic profile that applies to all of the data sets.

For information about protecting hierarchical file system (HFS) data in z/OS UNIX, see “Protecting HFS Data” on page 513.

Protecting General Resources

To protect a general resource, create a general resource profile using the RDEFINE command. When you create a general resource profile, you must specify a general resource class for the profile.

See “Appendix A. Description of RACF Classes” on page 565 for a list of the general resource classes that IBM supplies in the class descriptor table (CDT). The classes for z/OS and OS/390 systems are relevant to the system on which you are running Security Server (RACF). The classes for VM systems are primarily relevant if you share your RACF database with a VM system.

Installation-Defined Classes

Your installation can add new class descriptor table (CDT) entries or modify or delete existing entries that you have added in the installation-defined class descriptor table (ICHRRCDE).

When you define a new resource class, you can optionally designate that class as either a resource *group* class or a resource *member* class. For a resource group class, each user or group of users that is permitted access to that resource group is permitted access to all members of the resource group. Note that for each resource group class you create, you must also create a second class that represents the members of the group.

RACF refers to the class descriptor table (CDT) when it needs to make a class-related decision (such as, "What is the maximum length of profile names?"). With the CDT and appropriate use of RACF authorization checking services, you can extend RACF protection to any part of your system.

For more information on creating installation-defined classes, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Authority to Create Resource Profiles

Users can create data set profiles if any of the following is true:

- The high-level qualifier of the profile matches their user ID.
- A high-level qualifier matches a group in which they have CREATE authority.
- They have the SPECIAL or group-SPECIAL attribute.

Users can create general resource profiles if either of the following is true:

- They have the CLAUTH attribute for the class.
- They have the SPECIAL attribute.

Notes:

1. For complete descriptions of the authorizations required for any RACF command, see the description of the command in *z/OS SecureWay Security Server RACF Command Language Reference*.
2. If the SETROPTS GENERICOWNER option is in effect, further limitations apply. See "Restricting the Creation of General Resource Profiles (GENERICOWNER Option)" on page 117.

Authority to Modify or Delete Resource Profiles

To modify or delete a *generic* profile, you must meet at least one of the following criteria:

- Own the profile or, for data set profiles, have a user ID that matches the high-level qualifier of the profile name
- Have the SPECIAL attribute (or group-SPECIAL attribute, if applicable)

To modify a *discrete* profile, you must meet at least one of the following criteria:

- Own the profile or, for data set profiles, have a user ID that matches the high-level qualifier of the profile name
- Have the SPECIAL attribute (or group-SPECIAL attribute, if applicable)
- Have ALTER authority to the profile

Note: For complete descriptions of the required authorizations to any RACF command, or if adding members, see the description of the command in *z/OS SecureWay Security Server RACF Command Language Reference*.

Owners of Resource Profiles

In general, when you create a RACF profile, you become the owner of the profile unless you specify otherwise. You can choose to specify either a RACF group or a RACF-defined user ID:

- If you make a *user* the owner of the RACF profile, the user can modify, list, and delete the profile, or name another user to become the owner.
- If you make a *group* the owner of a RACF profile, you extend the scope of the group (and, in some cases, the scope of its superior groups) to the RACF profile. If users have the group-SPECIAL, group-AUDITOR, or group-OPERATIONS attributes in these groups, their authority extends to the new profile. Further, if the profile is a group profile, the scope can extend to profiles owned by the group itself.

For a list of the RACF commands that owners of resource profiles can issue, see Table 48 on page 581.

Note: The concept of ownership of any kind of RACF profile (user, group, or resource) is different from other kinds of ownership:

- When a user attempts to access a protected resource, the user might be considered an “owner” of the resource, and be given the equivalent of ALTER access authority. This is true, for example, when a user opens a data set whose high-level qualifier matches the user’s user ID.
- In data set profiles, you can specify a “resource owner” in the RESOWNER field. This field is used when users allocate new SMS-managed data sets protected by the profile. For more information, see “Determining the Owner of an SMS-Managed Data Set” on page 490.

Setting Up the Global Access Checking Table

You can use global access checking to improve the performance of RACF authorization checking for selected resources. For example, an entry in the global access checking table can allow all of the users on the system to have READ access to the ‘SYS1.HELP’ data set.

The global access checking table is maintained in storage and is checked early in the RACF authorization checking sequence. If an entry in the global access checking table allows the requested access to a resource, RACF performs no further authorization checking. This can eliminate the need for I/O to the RACF database to retrieve a resource profile, which can result in substantial performance improvements.

Note: If an entry in the global access checking table allows a requested access to a resource, no auditing is done for the request.

For information on planning and setting up the global access checking table, see “Setting Up the Global Access Checking Table” on page 206.

Security Classification of Users and Data

Security classification of users and data allows installations to impose additional access controls on sensitive resources. Each user and each resource can have a security classification in its profile. You can choose among the following:

- Security levels, security categories, or both.
- You can use security labels, which are a combination of security levels and security categories, and are easier to maintain.

For more information, see “Chapter 4. Classifying Users and Data” on page 95 and “B1 Security Level” on page 11.

Selecting RACF Options

RACF options provide flexibility in the creation and administration of your RACF security system. When implemented, RACF options can effectively enhance performance and recovery.

The SETROPTS command activates many of the RACF system-wide options. For a description of the SETROPTS options, as well as others, see “Chapter 5. Specifying RACF Options” on page 111.

Using RACF Installation Exits to Customize RACF

You can tailor RACF using various *installation exits* to bypass security checking or perform additional security processing or checking. (Installation exits are perhaps more in the realm of the technical support personnel and are discussed in detail in *z/OS SecureWay Security Server RACF System Programmer's Guide*. However, because you are responsible for overall security control at the installation, it is necessary for you to be aware of the use of installation exits.)

See *z/OS SecureWay Security Server RACF System Programmer's Guide* for complete information on coding RACF installation exits. To obtain a report that describes the RACF installation exits on your system, use the data security monitor (DSMON).

The RACROUTE REQUEST=VERIFY, VERIFYX, AUTH, and DEFINE Exits

The RACROUTE REQUEST=VERIFY, REQUEST=AUTH, and REQUEST=DEFINE macros perform user verification, access checking, and resource definition, respectively. Preprocessing installation exits are available to tailor the parameters specified by the REQUEST=VERIFY, REQUEST=AUTH, and REQUEST=DEFINE macros or to perform any additional security checks. Postprocessing exits are available to override or modify results of the RACF processing performed by these three macros. Because several of the RACF commands use REQUEST=AUTH and REQUEST=DEFINE when performing their functions, you can use the REQUEST=AUTH and REQUEST=DEFINE exits for some command tailoring.

The RACROUTE REQUEST=LIST Exits

The RACROUTE REQUEST=LIST macro, used to build in-storage copies of general resource profiles, has two exits: the preprocessing/postprocessing exit and the selection exit. RACF calls the preprocessing/postprocessing exit before RACROUTE REQUEST=LIST processing to allow the installation to alter REQUEST=LIST processing options and after REQUEST=LIST processing to perform housekeeping. RACF calls the REQUEST=LIST selection exit to resolve conflicts between new and existing profile information.

The RACROUTE REQUEST=FASTAUTH Exits

The RACROUTE REQUEST=FASTAUTH macro uses the profiles constructed in a data space by the RACROUTE REQUEST=LIST macro, and under certain circumstances by the SETROPTS RACLIST command, to perform authorization checking. REQUEST=FASTAUTH has exits that enable you to make additional security checks, or to instruct RACROUTE REQUEST=FASTAUTH to accept or fail the request to access a resource.

The RACROUTE REQUEST=FASTAUTH macro has two preprocessing and two postprocessing exits. The invocation of these exits is determined by the

circumstances involved in the invocation of the macro. These circumstances include whether the macro is invoked in cross-memory, and which operands are specified. See the RACROUTE REQUEST=FASTAUTH exit information in *z/OS SecureWay Security Server RACF System Programmer's Guide* for details.

The RACF Command Exits

RACF provides a command exit that is invoked before and after the execution of all RACF TSO commands except the block update command (BLKUPD), RACDCERT, RACLINK, and RVARY. This exit permits the installation to perform functions that include:

- Checking additional authorization
- Modifying authorization checking when these commands are issued
- Changing the options specified on the command
- Stopping the command from executing

There are other command exits that are invoked during processing of certain commands. See *z/OS SecureWay Security Server RACF System Programmer's Guide* for information on which exits are invoked for each command.

The RACF Password Processing Exit

The password processing exit, ICHPWX01, supplements the processing that RACF performs for new passwords and password change interval values. This exit gains control from the PASSWORD and ALTUSER commands and from RACROUTE REQUEST=VERIFY. The PASSWORD and ALTUSER commands and RACROUTE REQUEST=VERIFY call this exit before they actually change the current password or password change interval.

The RACF Password Authentication Exits

You can use the password authentication exits, ICHDEX01 and ICHDEX11, to control how RACF either encodes or masks the RACF password or OIDCARD data that is stored in the RACF database. By default, RACF encoding uses a software implementation of the data encryption standard (DES) algorithm and RACF masking uses a masking routine. You can use the password authentication exits to replace the RACF DES encoding routine with your own routine.

Tools for the Security Administrator

RACF provides a number of tools to help you monitor and control RACF events and manage the RACF database. These tools include:

- The RACF utilities
- The block update command (BLKUPD)
- The RACF report writer
- The data security monitor
- Commands to record statistics in discrete profiles
- The RACF LIST commands
- The RACF SEARCH command

Using RACF Utilities

RACF provides several utility that can help you and the RACF system programmer manage the RACF database and extract information from it:

Utility	Description
IRRMIN00	RACF database initialization utility

IRRUT400	RACF database split/merge/extend utility
IRRDBU00	RACF database unload utility
IRRUT200	RACF database verification utility
IRRUT100	RACF cross-reference utility
IRRRID00	RACF remove ID utility
IRRADU00	RACF SMF data unload utility

RACF Database Initialization Utility (IRRMIN00)

The RACF database initialization utility, IRRMIN00, formats a data set on DASD for use as the RACF database. You can use IRRMIN00 to initialize a new database or to update an existing RACF database with a new set of RACF templates.

For information about how to run IRRMIN00, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

RACF Database Split/Merge/Extend Utility (IRRUT400)

Using the RACF database split/merge/extend utility, IRRUT400, you can split a single RACF database into two or more data sets, merge two or more data sets of the RACF database together into one or more data sets, or copy a RACF database from one type of device to another. In the process, IRRUT400 physically reorganizes the RACF profiles and compresses the RACF database.

For information about how to run IRRUT400, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

RACF Database Unload Utility (IRRDBU00)

The RACF database unload utility, IRRDBU00, gives you easy access to the information in your RACF database. You can use IRRDBU00 to unload your RACF database to a sequential data set and then use the output in a variety of ways. For example, you can view the data set directly, sort it, merge it with other data, or use it as input to your own analysis and reporting programs. In addition, you can upload the sequential data set to a database manager such as DB2, where you can easily query the data and create reports. For information about how to run IRRDBU00 and use its output effectively, see "Using the RACF Database Unload Utility (IRRDBU00)" on page 322.

RACF Database Verification Utility (IRRUT200)

You can use the RACF database verification utility, IRRUT200, to identify inconsistencies in the internal organization of a RACF database. You can also use it to make an exact, block-by-block copy of the RACF database.

For information about how to run IRRUT200, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

RACF Cross-Reference Utility (IRRUT100)

The RACF cross-reference utility, IRRUT100, lists all occurrences of a specified user ID or group name that appear in a RACF database. This can help you discover the relationships between various users and groups, and learn other important information about users, groups, and the resources they control. For example, you can use the output to verify that users have the right access to resources. You can also use the output to determine the resources whose ownership must be transferred before you delete a user or group from the RACF database.

All users can run IRRUT100 for their own user IDs or any user IDs they own. To run IRRUT100 for other users' IDs, you must be defined to RACF with one of these attributes:

- AUDITOR
- SPECIAL
- group-AUDITOR
- group-SPECIAL

IRRUT100 produces a cross-reference report that describes the occurrences of each user ID or group name that you specify. Generic profile names are followed by the letter "G" in parentheses.

For information about how to run IRRUT100 and use its output, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

As an alternative to IRRUT100, you can use the output from the database unload utility (IRRDBU00). For more information, see "Using the Database Unload Utility Output Effectively" on page 326.

RACF Remove ID Utility (IRRRID00)

The RACF remove ID utility, IRRRID00, can help you keep the RACF database current. You can use this utility to remove all references to group names and user IDs that no longer exist in or are about to be removed from the RACF database. Also, you can specify a replacement ID for those IDs that will be removed.

The remove ID utility processes the output of the RACF database unload utility (IRRDBU00). You can use the remove ID utility to:

1. Find all residual references to user IDs and group names that no longer exist in the RACF database.
2. Find all references to a list of user IDs and group names that you specify in the SYSIN file.

For information about how to run IRRRID00 and use its output see "Using the RACF Remove ID Utility (IRRRID00)" on page 342.

RACF SMF Data Unload Utility (IRRADU00)

You can create a sequential file from security-related audit data using the RACF SMF data unload utility, IRRADU00. Once you create the sequential file, you can use it in a variety of ways. For example, you can view the file directly, sort it, merge it with other data, or use it as input to your own analysis and reporting programs. In addition, you can upload the sequential file to a database manager such as DB2, where you can easily query the data and create reports.

Note: You can use this utility to create reports for RACF audit records that the RACF report writer is unable to process.

For information about how to run IRRADU00, see *z/OS SecureWay Security Server RACF Auditor's Guide*.

RACF Block Update Command (BLKUPD)

You can repair your RACF database by directly changing its internal elements (blocks) using the RACF block update command (BLKUPD). Because this direct manipulation can result in an inconsistent and therefore unusable database

structure, you must be very careful when using BLKUPD. For assistance in using BLKUPD, contact your system programmer or the IBM support center.

For more information on BLKUPD, see *z/OS SecureWay Security Server RACF Diagnosis Guide*.

Using the RACF Report Writer

The RACF report writer lists information contained in RACF-generated SMF records. With the RACF report writer you can:

- Collect data about successful accesses and warnings before building resource profile access lists.
- List the contents of RACF SMF records in a format that is easy to read.
- Obtain reports that describe attempts to access a particular RACF-protected resource. These reports contain the user ID, the number and type of successful accesses, the number and type of unauthorized access attempts, and the name of the user if available in the SMF record.
- Obtain reports that describe user and group activity.
- Obtain reports that summarize system and resource use.

The output from the RACF report writer includes a header page that explains the meaning of the event and qualifier numbers that appear in the SMF record listings and summary reports. The remainder of the report comes in various forms, according to your selection. You can request a general summary, SMF record listings, and summary reports.

The RACF report writer has been stabilized at the RACF 1.9.2 level and has not been enhanced for this release. For example, it does not support audit records for z/OS UNIX events. To create reports and use the audit records for those events, use the SMF data unload utility.

See *z/OS SecureWay Security Server RACF Auditor's Guide* for complete information on using the report writer.

Using the Data Security Monitor

The data security monitor (ICHDSM00, usually called DSMON) is a batch program that allows authorized users to obtain a set of reports that provide information about the current status of your installation's data security environment.

These reports help you (1) check the initial steps you took to establish system security and (2) make additional security checks periodically. If the DSMON program (ICHDSM00) is defined in the PROGRAM class (that is, it is a controlled program), the user must be authorized through its access list before the program can be run. If DSMON is not a controlled program, the user must have the AUDITOR attribute to run DSMON and produce reports. (For more information on controlled programs, see "Program Control" on page 250.)

For more information on the DSMON reports, see "Using the Data Security Monitor (DSMON)" on page 310 or *z/OS SecureWay Security Server RACF Auditor's Guide*.

Recording Statistics in RACF Profiles

In addition to placing statistical information into the various profiles when you create them, you can cause RACF to dynamically record statistics (such as the number of

user accesses to a protected resource) in discrete profiles. For data sets and general resource classes, you can optionally record the following statistics:

- The number of times that a resource that is protected by a discrete profile was accessed under a specific RACF authority level (such as READ or UPDATE).

  Note: When a RACROUTE REQUEST=AUTH is accepted at a certain level of authority (such as UPDATE), this does not necessarily mean that data is actually updated.
- The number of times that a specific user or group accessed a resource protected by a discrete profile.
- The date when a resource profile was last updated.

These statistics enable you to monitor the current operation of your computing system for administrative and control purposes. You can list the statistics and other descriptive information that is recorded in RACF profiles with various RACF commands.

Note: Statistics are not recorded for either of the following types of profiles:

- Generic profiles
- In-storage profiles (in classes for which the SETROPTS RACLIST command or RACROUTE REQUEST=LIST has been issued)

Listing Information from RACF Profiles

You can list the contents of profiles by using the LIST commands, as shown in Table 2.

Command output is in line mode unless you use ISPF panels. To capture the output of RACF commands, do the following:

- Use the TSO session manager to scroll through the output.
- Submit a batch TMP job that issues RACF commands and save the held output.

As an alternative to the LIST commands, you can obtain this information from the output of the database unload utility, IRRDBU00. See “Using the Database Unload Utility Output Effectively” on page 326.

Table 2. Commands to List Profile Contents

Command	Function
LISTDSD	Lists the contents of data set profiles and lets you determine which generic profile applies to a particular data set. For a description of how to do this, see *z/OS SecureWay Security Server RACF General User's Guide*. The listing shows: • Owner of the profile • UACC • Date the profile was created • Users and groups that are authorized to access the data set • Your highest access authority to the data set the security label (if there is one), and other information • Other information See *z/OS SecureWay Security Server RACF General User's Guide* for a complete description of this listing.

Table 2. Commands to List Profile Contents (continued)

Command	Function
LISTGRP	Lists the contents of group profiles. The listing shows: • Owner of the group profile • Superior group name • Users connected to the group • Subgroup names • Other information
LISTUSER	Lists the contents of user profiles. The listing shows: • Owner of the profile • User name • Default group name • Groups that a user is connected to • Group authorities • Date the password was last changed, • Default security label • Other information See *z/OS SecureWay Security Server RACF General User's Guide* for a complete description of this listing.
RACDCERT LIST	Lists digital certificate information. For each digital certificate, the listing shows: • Serial number • Issuer's distinguished name • Status information • Up to 256 bytes of the subject's name • Label • Date and time from which the certificate is valid • Date and time after which the certificate is no longer valid • Private key type • Private key size • Key ring associations, including ring name and ring owner For an example of this listing, see Figure 56 on page 526.
RACDCERT LISTRING	Lists key ring information. For each digital certificate in the ring, the listing shows: • Ring name • Label assigned to the certificate • Owner of the certificate (ID(*name*), CERTAUTH or SITE) • Usage within the ring • DEFAULT status within the ring For an example of this listing, see Figure 57 on page 527.

Table 2. Commands to List Profile Contents (continued)

Command	Function
RACDCERT LISTMAP	Lists certificate name filter information. For each filter, the listing shows: • Label assigned to the certificate name filter • Trust status • Issuer's name filter, if applicable • Subject's name filter, if applicable • Criteria information, if applicable For examples of this listing, see Figure 60 on page 535 and Figure 67 on page 542.
RACLINK LIST	Lists user ID associations. For each association, the listing shows: • Type of association • Node and user ID • Whether password synchronization is enabled • Status of the association For an example of this listing, see "Listing User ID Associations" on page 363.
RLIST	Lists the contents of profiles for general resources such as tape volumes, DASD volumes, IMS transactions, and terminals. The listing shows: • Owner of the profile • UACC • Date the profile was created • Users and groups that are authorized to access the resource • Your highest access authority to the resource • Security label • Other information

Searching for RACF Profile Names

You can list the names of profiles that meet certain search criteria by using the SEARCH command. This command is described in Table 3 on page 32.

Note: The output of this command is in line mode unless you use ISPF panels. You can use the TSO session manager to scroll through the output from the listing commands.

As an alternative to the SEARCH command, you can use the RACF database unload utility (IRRDBU00) to find additional data. The output from the database unload utility used with a relational database manager can provide you with the ability to implement additional search criteria.

Table 3. Commands to Search for Profile Names

Command	Function
SEARCH	Searches the RACF database for the names of profiles (in a particular resource class) that match the criteria you specify. For example, you can search for all TERMINAL profiles that have a security level specified. You can save the list of profile names in a data set. You can easily specify RACF commands (or other commands) to be saved with the profile names, generating a CLIST that you can run against the profiles.

The search criteria can include one or more of the following:

- Profile names that contain a specific character string
- Profiles for resources that have not been referenced for more than a specific number of days
- Profiles that contain a level equal to the level you specify
- Profiles with the WARNING indicator
- Profiles that contain a specific security level, category, or label
- Profiles to which another user has access

Note: Unless you have the SPECIAL attribute, you must have at least READ access authority for each profile whose name is listed as the result of your request.

Using the LIST and SEARCH Commands Effectively

Attention

Using the SEARCH command can slow the system's performance. Therefore, SEARCH should be used with discretion (or not at all) during busy system times.

You may want to investigate using the database unload utility for some of your profile searches. The database unload utility need not slow the system's performance and, in some cases, can provide the same information as the SEARCH command.

Question: How can I tell whether (or how) a data set is protected?

Answer: The answer is complicated by a number of factors, including the presence of discrete and generic data set profiles, whether the data set is RACF-indicated, and the setting of such system-wide options as SETROPTS GENERIC(DATASET) and SETROPTS PROTECTALL. For more information, see "Protecting Data Sets" on page 150.

Question: How can I tell if (or how) a resource (other than a data set) is protected?

Answer: Use the RLIST command, omitting both the GENERIC and NOGENERIC operands:

```
RLIST classname resource-name
```

Note: For resources that have grouping classes (such as terminals, DASD volumes, and certain IMS and CICS classes), specify the related "member class" and the RESGROUP operand on the RLIST command:

```
RLIST member-class resource-name RESGROUP
```

For example, for terminal T1:

```
RLIST TERMINAL T1 RESGROUP
```

This lists the profiles in the GTERMINL class that protect terminal T1.

This example does not work for terminals protected by a generic member in the GTERMINL class.

Question: How can I find the data sets that a user can access?

Answer: Take the following steps:

1. Find the names of the profiles the user has access to:

   ```
   SEARCH USER(userid) NOMASK
   ```

   The name of a discrete profile identifies which data set it protects.
2. For each generic profile whose name is listed in step 1, list the cataloged data sets protected by the profile (assumes that the SETROPTS CATDSNS option is in effect):

   ```
   LISTDSD DATASET(generic-profile-name) DSNS NORACF
   ```

   Note: To find out *how* a user can access a particular data set (READ, UPDATE, and so forth), analyze the profile protecting the data set to determine how RACF authorization processing would respond to an access request.
3. Find the entries in the global access checking table for the DATASET class:

   ```
   RLIST GLOBAL DATASET
   ```

   These entries allow all users access to data sets that match.

Question: How can I find the general resources that a user can access?

Answer: This must be done one class at a time. For each class, take the following steps (which are similar to the steps for data sets):

1. Find the names of the profiles the user has access to:

   ```
   SEARCH CLASS(classname) USER(userid)
   ```

   The name of a discrete profile identifies which resource it protects.

   Notes:

   a. If the resource is in a class for which there can be resource group profiles (such as GTERMINL, GDASDVOL, and so forth), issue the SEARCH command twice, once for the member class and once for the grouping class.

      For example, for terminals:

      ```
      SEARCH CLASS(TERMINAL) USER(userid)
      SEARCH CLASS(GTERMINL) USER(userid)
      ```

   b. If the SEARCH command shows a profile that contains a RACF variable (indicated by one or more ampersands (&) in the name), you must list the RACFVARS profile that defines the variable. For example, if you see a profile named SAMPLE.&X.DATA, use the RLIST command to list the RACFVARS profile that defines the variable:

      ```
      RLIST RACFVARS &X
      ```

2. RACF provides no direct way to determine which resources a particular general resource profile protects, as in issuing the LISTDSD command with the DSNS operand. This is because there is not generally a list, stored on the system, of the various existing resources that RACF can check. There would have to be such a list for each general resource class—and there are well over 50 resource classes (from terminals to JES input devices to tape volumes). Thus, for any particular class, an auditor or administrator would have to consult with the profile owners (or system support) to determine exactly which resources a generic profile protects.
3. Find the entries in the global access checking table for the class:

   ```
   RLIST GLOBAL classname
   ```

   These entries allow all users access to data sets that match.

Question: How can I find the user or group profiles a user can list or alter?

Answer: Enter one of the following commands:

```
SEARCH CLASS(USER) USER(userid)
SEARCH CLASS(GROUP) USER(userid)
```

Question: How can I find out the members of a RACF group?

Answer: Enter the following command:

```
LISTGRP groupname
```

Question: How can I find out what groups a user belongs to?

Answer: Enter the following command:

```
LISTUSER userid
```

See *z/OS SecureWay Security Server RACF General User's Guide* for more detail on the output of these commands.

Chapter 2. Organizing for RACF Implementation

This chapter outlines the planning that your installation should do before you install RACF.

This book describes the security administrator's tasks as they relate to RACF. A successful security program, however, goes well beyond the relationship of the security administrator to the software security program your company has chosen to protect its computerized data. This chapter discusses some of the early work you and other people must do before installing RACF.

Ensuring Management Commitment

Management's decision to install RACF is not, by itself, enough to ensure adequate security at your location. Indeed, if management were to ignore security concerns after simply selecting *any* software protection package, the eventual result would most likely be the failure of the security undertaking.

To be successful, a security implementation requires a management that is involved with questions of security policy and procedures, the resources to be allocated to the security function, and the accountability of users of the computer system. Without such management support, the security procedures will fall into disuse and become more of an administrative chore than a viable protection scheme. (In fact, such a situation could breed a false sense of security that could lead to serious exposures.)

You should work with management to prepare a clear, inclusive statement of security policy. This statement should reflect:

- Corporate security policy
- Physical protection considerations
- Installation data processing security requirements
- User department security requirements
- Auditing requirements
- A statement of policy concerning outside users of the system
- The security attitudes that will be expected from all users of the system

The resultant security policy helps to ensure that a security implementation team can prepare a RACF implementation plan that is both realistic and consistent with the installation's security policy.

Selecting the Security Implementation Team

To ensure a smooth implementation of RACF, careful planning is required, starting with your selection of an implementation team.

The implementation team should include the viewpoints of all of the user types (security and group administrators, auditor, technical support personnel, operations, and end user). In addition to knowing their own areas, the implementation team representatives should be familiar with, or have access to people who are familiar with, the following areas:

- RACF
- Privacy legislation
- The installation's organization
- Installation standards
- Major application areas

As security administrator, you should lead the implementation team. For best results, you should keep the team as small as possible. You should ensure that the results of the team's work are reviewed and fully supported by management.

Responsibilities of the Implementation Team

Some of the responsibilities that might be assigned to the implementation team are:

- Defining RACF security objectives
- Deciding what to protect and how to report attempted violations
- Establishing resource ownership structures
- Developing the RACF implementation plan and installing RACF
- Educating all users of the RACF-protected system

Table 4 describes the responsibilities of typical implementation team members.

Table 4. Participants of the Security Implementation Team

User Type	Responsibility
Security Administrator	As security administrator, you have overall responsibility for RACF implementation. It is your job to ensure that the work of the implementation team is consistent with good security practice and in line with the security policy established earlier. In addition, you or your delegate administrators should be responsible for educating the installation users about how RACF will be implemented. (That is, will there be a grace period before the new security procedures take effect? How will the implementation of RACF affect the day-to-day responsibilities of each user?)
Technical Support Person	The technical support person is normally a system programmer who installs RACF and maintains the RACF database. This person has overall responsibility for the programming aspects of system protection and provides technical input on the feasibility of implementing various aspects of the implementation plan. In addition, the technical support person writes, installs, and tests RACF exit routines, if they are required. If you will have RACF installed on more than one system in your installation, the implementation team should include a technical support person for each system on which you are using RACF. For more information, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Table 4. Participants of the Security Implementation Team (continued)

User Type	Responsibility
Auditor	The auditor provides guidance on good auditing practice as it relates to data security and user access. This person implements the necessary RACF logging and reporting options to provide an effective audit of security measures. For more information on the auditor's duties, see *z/OS SecureWay Security Server RACF Auditor's Guide*.
User Representative	The user representative should be a prospective group administrator who represents a major application area—perhaps a user support services or liaison function.
Other Users	Other users might be considered as members of the implementation team if appropriate. For example, other users who are involved with security include CICS, TSO, and database administrators and JES, MVS, and PSF system programmers.

The rest of this chapter discusses some of the major responsibilities of the security implementation team.

Defining Security Objectives and Preparing the Implementation Plan

Working from the statement of security policy as a base, the implementation team prepares an *implementation plan*. This plan should answer the question "How do we get there from here?" Experience indicates that an evolutionary implementation of security, rather than a revolutionary one, is the most successful way to bring about adequate security measures in the quickest time possible.

The implementation team needs to set priorities about the data, applications, and users that must be secured. The implementation team should plan to phase in the security controls over a period of time to give users time to adjust to them.

The implementation plan should identify the major RACF events—when each must be completed, who is responsible for each event, and interdependencies among events. In addition, the plan should take into account any other significant activity planned for the same time period that could affect the implementation (for example, new systems, hardware, and applications). At an early stage the team should also define a pilot group for whom the protection of business data, jobs, and users will be completed before undertaking the protection of other business data. The pilot group provides a means of obtaining RACF experience before extending protection to the rest of the installation.

Deciding What to Protect

Every installation has varying amounts of confidential data and varying degrees of confidentiality. For example, a development laboratory might be primarily concerned with the confidentiality of new products, whereas a bank or an insurance agency would be concerned with the confidentiality of its customers' records. Generally speaking, though, all data falls into one of the following categories:

1. Very sensitive, confidential data, which requires protection from disclosure, modification, or destruction
2. Non-confidential data, which is recoverable with little inconvenience if destroyed
3. Data that falls between these two extremes, which should be protected from inadvertent or deliberate modification or destruction

Most data falls into the last category.

Obviously, the data in the first category *must* be protected. What should also be considered is how to protect the data that *ought* to be protected in a simple yet effective manner—in a way that is transparent to the user of this data. The implementation team does a *risk evaluation* of the installation's data to determine which data needs what level of protection.

The task of protecting large quantities of data can take on significant proportions unless you can acquire this protection automatically. In the case of RACF, protecting data is quite simple and, after the controls are in place, practically free from administrative overhead.

Protecting Existing Data

To protect data that already exists on your system before RACF is installed, you must create RACF profiles. You can use either discrete or generic profiles. However, using generic profiles can reduce the administrative effort of this task, because one generic profile can protect many resources. For example,

- You can protect existing data sets by using the ADDSD command. You should consider creating at least one profile for each high-level qualifier (user ID or group name) on your system. You can specify profile names with the format:

  `'high-level-qualifier.*'`

  If enhanced generic naming is in effect, use:

  `'high-level-qualifier.**'`

  Note: You must determine the appropriate UACC, access lists, and other information (such as security classification, if used) for each profile.

  For resources that have unique security requirements, you must create discrete profiles.
- You can also protect existing general resources (such as tape volumes or terminals) by using the RDEFINE command. If several resources in the same class have the same access requirements, you can use one profile to protect them. Not only does this save space, but it also saves administrative time.

  If the names of the resources contain some identical characters, you can usually create generic profiles whose names contain the *, **, or % character to protect the resources.

  For certain classes, such as terminals, DASD volumes, and others, you can create resource grouping profiles to protect resources whose names do not lend themselves to the use of the *, **, or % character.

  For any general resource class, you can define a "RACF variable" that can be used in the profile names in general resource classes. For more information about how to select the type of profile to protect a resource, see "Choosing Among Generic Profiles, Resource Group Profiles, and RACFVARS Profiles" on page 199.

Note: You must determine the appropriate UACC, access lists, and other information (such as security classification, if used) for each profile.

For resources that have unique security requirements, you must create discrete profiles.

Protecting New Data

RACF provides several ways to protect new data:

- **Generic Profiles:** Use of generic profiles can decrease the amount of administrative effort because you can use a single generic profile to protect a large number of existing resources that have a similar naming structure. Generic data set profiles protect existing data sets even if they are not RACF-indicated. (See “Choosing between Discrete and Generic Data Set Profiles” on page 157 and “Protection through Generic Profiles” on page 156.)
- **Automatically-Created Discrete Profiles:** RACF automatically protects new data sets by creating a discrete profile for each data set when the user creating them has the ADSP attribute or has specified the PROTECT=YES operand on the JCL DD statement that creates the data set. This automatic definition of discrete data set profiles occurs when the resource manager issues RACROUTE REQUEST=DEFINE.

  Notes:

  1. ADSP and PROTECT=YES always cause the creation of a discrete profile, which is desirable for data sets that have unique access-authorization requirements. If your data sets do not have unique access-authorization requirements, consider using generic profiles.
  2. By themselves, ADSP and PROTECT=YES allow only the creator of the data set to access the protected data. One way to allow other users access to the protected data set is to use the PERMIT command to place them (or groups of which they are members) on the access list of the profile with the desired access authority. Also, if the data set being created is a group data set, and the user creating it has the GRPACC attribute in that group, all members of the group are given UPDATE access authority to the group data set.
- **Automatic Profile Modeling:** One way you can allow other users to access protected data is by using *automatic profile modeling*. When you use automatic profile modeling, the profile that protects a new user or group data set automatically has an access list copied from the model profile. Therefore, users defined in the access list of the model can access the newly-created user or group data set. Automatic modeling is thus valuable for establishing the initial access list for newly-created generic data set profiles. You can use automatic profile modeling for profiles that are created by the user’s ADSP attribute, the PROTECT=YES operand of the JCL DD statement, or the ADDSD command.

Profile Modeling

Profile modeling enables RACF or an installation exit routine to copy information (such as the access list, owner, and logging options) from an existing (model) profile when defining a new profile. (The copied profile is not necessarily identical to the model profile. For a description of the differences, see “Possible Changes to Copied Profiles When Modeling Occurs” on page 40.) This copying greatly reduces the effort needed to create new profiles. Some examples of using profile modeling are:

- A user can copy information from an existing profile into a new profile by using the FROM operand (and related operands) on the ADDSD or RDEFINE commands. RACF uses the specified profile as a model when creating the new profile. However, profile segment information (CICS, DFP, DLFDATA, LANGUAGE, OPERPARM, SESSION, TSO, WORKATTR, and so on) is not copied to the new profile.
- A user can copy the access list from an existing profile into another existing profile using the FROM operand (and related operands) on the PERMIT command.
- For data sets, an installation can use automatic profile modeling. A user with the SPECIAL attribute can specify MODEL(USER), MODEL(GROUP), or

MODEL(GDG) on the SETROPTS command. These operands specify that RACF is to use a model data set profile for selected users, groups, or GDG data sets.

If the SETROPTS MODEL options are in effect, the MODEL operands of the ADDUSER, ADDGROUP, ALTUSER, and ALTGROUP commands specify the data set profile that is to be used as a model from which to copy information into new data set profiles.

For more information on this topic, see “Automatic Profile Modeling for Data Sets” on page 163.

- If the preceding methods are not sufficient, an installation can also use a REQUEST=DEFINE exit routine to supply either the name of a model profile or the profile itself.

Possible Changes to Copied Profiles When Modeling Occurs

When a profile is copied during profile modeling, the new profile could differ from the model in the following ways:

- RACF places the user creating the new profile on the access list with ALTER access authority or, if the user is already on the access list, changes the user's access authority to ALTER. This is true only if ADDCREATOR is in effect, or if you are creating a discrete DATASET or TAPEVOL profile with RACROUTE REQUEST=DEFINE. Otherwise, the user creating the new profile is not placed on the access list or, if the user is already on the access list, the user's authority is not changed when the access list is copied to the new profile.

  If the profile being added is for a group data set and the user has the GRPACC attribute for that group, RACF places the group on the access list with UPDATE access authority or, if the group is already on the access list, changes the group's access authority to UPDATE.

  Note: These access list changes do not occur if the data set profile is created only because the user has the OPERATIONS attribute.
- If the model profile contains members (specified with the ADDMEM operand), the members are not copied into the new profile.
- If the SETROPTS MLS option is in effect, the security label, if specified in the model profile, is *not* copied. Instead, the user's current security label is used. For more information on security labels, see “Understanding Security Labels” on page 101.

  Note: When the SETROPTS MLS option is in effect, if the SETROPTS MLSTABLE option is also in effect and the user has the SPECIAL attribute, the security label specified in the model profile is copied to the new profile. For more information on security labels, see “Understanding Security Labels” on page 101.
- For TAPEVOL profiles, TVTOC information is not copied to the new profile.
- Even if SETROPTS NOADDCREATOR is set, the model profile access list is copied exactly. Therefore, if the creator's user ID appeared in the model's access list, the authority is copied to the new profile exactly.

Allowing a Warning Period

In addition to deciding what to protect, the implementation team must consider how to phase in the new security controls with minimum disruption of current work patterns. You should consider:

- Auditing all accesses allowed by a resource profile
  - Specify GLOBALAUDIT(ALL) for the resource profile.
- Auditing all protected resources in a class

– Enter the SETROPTS LOGOPTIONS command.

These commands cause SMF logging to occur for all accesses. If the profiles allow all access, the SMF records indicate what users or jobs need access to the protected resources.

RACF also provides the option of issuing a warning message to users instead of failing a request to access a resource. You can control which resources are protected in this manner by specifying the WARNING operand on the ADDSD, RDEFINE, ALTDSD, or RALTER command. When a resource check is performed, if the check fails and WARNING has been specified, RACF issues a warning message to the user, logs the access, and allows the user to access the resource.

Notes:

1. The warning message facility applies to in-storage profiles created by the SETROPTS RACLIST command. It may or may not apply to in-storage profiles created by the RACROUTE REQUEST=LIST macro, depending on the options chosen by the resource manager that issues the macro.
2. The warning message is issued for existing data sets, not for new data sets. For example, if you try to create a new data set that has the same name as a generic profile that does not give you ALTER access, the create fails if this profile does not have WARNING set on. If, however, WARNING is on in the generic profile, the data set allocation proceeds but no warning message is issued.

Establishing Ownership Structures

RACF provides enough flexibility so that in most cases your RACF ownership structures can correspond to your existing installation management and organizational structures. However, this flexibility does not mean that some realignment of the organizational structures might not be advantageous from the security standpoint.

In any event, you should subdivide the ownership structures to minimize both occasions when data needs to be passed between groups, and occasions when exceptional access controls are required. If you define groups so that all users in a group share common access requirements, your administrative task of authorizing users is greatly simplified.

Selecting User IDs and Group Names

In your installation it might be enough for you to isolate development work from production. On the other hand, it might be more practical for you to define many individual users and groups. In either case, you should take a look at what already exists and modify RACF to adapt to the current environment. For example, do any or all of the system users already have user IDs? If so, perhaps you can make use of them. For example, every data set name has its owner's user ID as its high-level qualifier by default.

Batch Users: Batch users might not already have user IDs. Here, you might consider assigning user IDs based on personnel number or, if appropriate, group name. If it is not clear what to use as a user ID, start by considering group names. Again, examine what already exists:

1. Is there an existing organizational structure that has groups with suitable abbreviations? Can the existing structure be used as is, or modified to suit?

2. What conventions already exist in job statements? It is common for the first few characters of the job names to be meaningful in terms of an application name, project, department, or some other such functional group. Could these be used as group names, or even a user ID? Are there any other fields in the job statement that could be used (for example, the account number or programmer name)? That is, could you determine from a job statement to whom or to what functional group the job belongs? (Note: The ability to derive a user ID or group from existing job statement information can be a significant migration aid. It could help you avoid the administrative effort of adding the USER=operands to existing job statements.)
3. Look at data set names to determine the local naming conventions for data sets. Can you determine to which functional group a data set belongs by looking at the name? Can you say “This is an IMS database,” or “This data set belongs to the payroll group”?

   It is likely that several naming conventions already exist. RACF options enable you to handle most existing variations.

Whatever you choose, consider carefully the longer term security objectives. Adding new groups and users to an existing structure presents few administrative problems. Even deleting users and groups can be done without much difficulty. However, a major reassignment of user IDs and group names, although possible, is best avoided by careful initial selection.

Establishing Your RACF Group Structure

You should map your groups to your organization’s structure and arrange them hierarchically so that each group is a subgroup of some other group. The group SYS1 is predefined as the highest group in the hierarchy. You should document the resulting group structure as part of the implementation plan. You might want to develop a set of guidelines for your delegated security and group administrators to identify the general categories of resources and users, and the relationships between them.

For groups that may become large, and for which a quick listing of members is not needed, you may wish to consider defining the groups using the UNIVERSAL operand of the ADDGROUP command. See “Defining Large Groups with the UNIVERSAL Attribute” on page 54.

Figure 4 on page 43 shows relationships that can exist between users and groups.

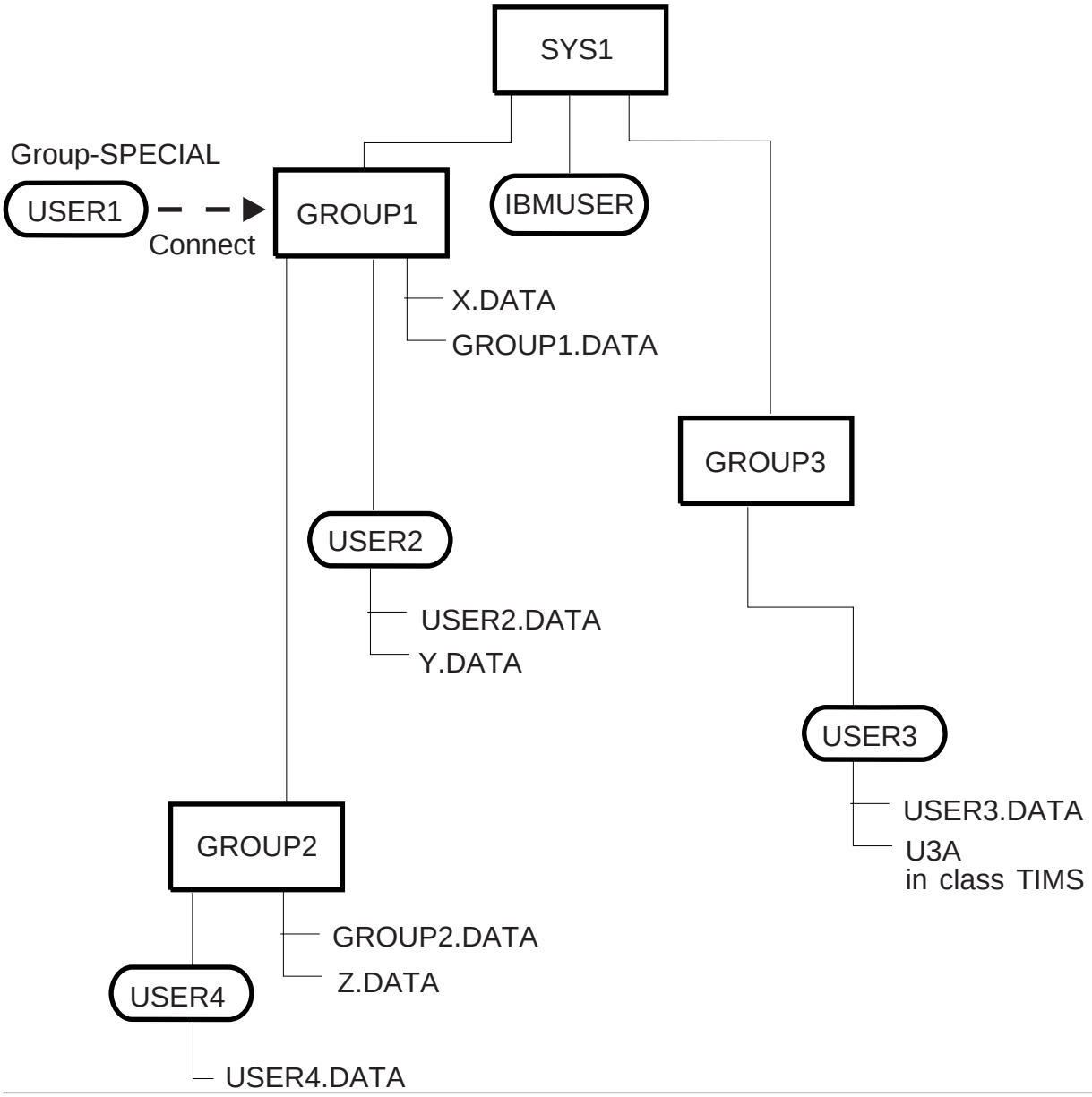


Figure 4. User and Group Relationships

In Figure 4:

- The users' default groups are their owning groups.
- Groups X and Y exist and are owned by GROUP1.
- Group Z exists and is owned by GROUP2.
- The highest level group, SYS1, owns subgroups GROUP1 and GROUP3 and the user IBMUSER.
- GROUP1 owns subgroup GROUP2 and the users USER1 and USER2.
- USER1 is connected to GROUP1 with group-SPECIAL authority. This gives USER1 (who is a RACF administrator) control over GROUP1's resources and resources in GROUP1's scope, but not over GROUP3's resources.

Note: If you run with list-of-groups checking inactive (that is, with the SETROPTS NOGRPLIST option in effect), the scope of USER1's group-SPECIAL attribute is limited to his default group or the group he specified when logging on, and the groups below that group in the hierarchy. For more information on list-of-groups checking, see "Activating List-of-Groups Checking (GRPLIST Option)" on page 116.

Educating the System Users

Part of your job is to tell the system users what they need to know to work without disruption when RACF is installed.

The amount of detailed information that each user needs to know about RACF depends on the RACF functions that you authorize the person to use. Here are some examples of information that various system users typically require:

All System Users: All users who are defined to RACF must know:

- How to identify themselves to the system with their user ID and password, and how to change their password. They should also be aware of the significance of their password to system security.
- If list-of-groups processing is *not* in effect, how to log on to a group other than their default group.

  Note: Users can use the LISTUSER command to find out the groups to which they belong.
- If security labels are used on your system, how to log on with a security label other than their default security label. For more information, see "Understanding Security Labels" on page 101.

  Note: To find out what security labels they can use, users can enter:

  ```
  SEARCH CLASS(SECLABEL)
  ```
- If you want them to be able to change their password intervals, how to use the PASSWORD command.
- How to use the LISTUSER command to list their own profile information.

For complete information on tasks general users can perform using RACF (such as permitting others to use their data sets), see *z/OS SecureWay Security Server RACF General User's Guide*.

Users of RRSF Functions: RRSF users need to understand RRSF network concepts and know RRSF node names. Depending on your security plan, some RRSF users might also need to know how to:

- Direct commands
- Synchronize passwords
- Establish and approve user ID associations using the RACLINK command.

For examples and more information, see *z/OS SecureWay Security Server RACF General User's Guide*.

Users Who RACF-Protect General Resources: Depending on your security plan, users might work with profiles in the TAPEVOL, JESSPOOL, or other general resource classes. These users must know:

- How to define and modify profiles in the general resource class, including whether generic profiles are allowed in the class

- What user IDs and group names they can use when giving access to the profiles
- The meaning of the access authorities (such as NONE, READ, and WRITE) in the general resource class
- What your installation's security policy is towards specific security enhancements like security levels, categories, and security labels

In addition to the education needed for administrators who are using generic profiles, even more education is required on generic profiles for those who are switching to enhanced generic naming (that is, from the SETROPTS NOEGN to the SETROPTS EGN option).

For more information, see "Defining Profiles for General Resources" on page 196 and the sections of this book that describe how to use the class.

Technical Support Personnel: Users who install the RACF component of the Security Server must be familiar with migration considerations and the steps that are required to install or reinstall RACF. For more information, see *z/OS SecureWay Security Server RACF Migration*.

Users who maintain the RACF database must be familiar with the RACF utilities, which are described in *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Group Administrators: Group administrators either have one of the group authorities, have a group attribute (such as group-SPECIAL), or own group resources. These users need to use the information in this book and *z/OS SecureWay Security Server RACF Command Language Reference*.

RACF Auditors: Users with the AUDITOR attribute should see *z/OS SecureWay Security Server RACF Auditor's Guide* for information on using RACF for auditing.

Note that if ISPF and TSO/E are installed, the user can use the RACF ISPF panels to perform the same functions as the RACF commands. Using the RACF ISPF panels frees users from the need to know the details of command syntax.

Note: You can ask a user with the AUDITOR attribute to issue the SETROPTS command with the CMDVIOL operand. This causes RACF to log all of the RACF command violations that it detects. The auditor can then use the RACF report writer to produce a printed audit trail of command violations. From the report, you can determine how many command violations are occurring and which users are causing the violations. A significant number of command violations, especially when RACF is first installed, may mean users need more education. The report can also help you to identify any specific users who are persistently trying to alter profiles without the proper authority.

z/OS SecureWay Security Server RACF Command Language Reference contains detailed information on the RACF commands used.

Programmers Writing Unauthorized Applications: Programmers writing unauthorized applications can use the RACROUTE macro to request many security-related services, including controlling access to protected resources (RACROUTE REQUEST=AUTH).

Note: Your installation can create installation-defined resource classes. If your installation creates profiles in those classes, an application can issue a RACROUTE REQUEST=AUTH to check if a user has sufficient authority to complete a user action. How much authority is needed for any particular user action is defined by the way in which the application invokes the RACROUTE REQUEST=AUTH macro. For more information on creating installation-defined classes, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Programmers Writing Authorized Applications: Programmers writing authorized applications (that is, APF-authorized programs) can use the RACROUTE macro to request security-related services, including:

- Identifying and verifying users (RACROUTE REQUEST=VERIFY)
- Replacing or retrieving fields in RACF profiles (RACROUTE REQUEST=EXTRACT)

For more information on using the RACROUTE macro, see *z/OS SecureWay Security Server External Security Interface (RACROUTE) Macro Reference*.

Summary

As an overall strategy in organizing for RACF implementation, the implementation team should strive for a policy of security by evolution, rather than revolution. Wherever transparency can be used, it should be. In some cases, you must actively solicit management support.

You should examine organizational structures to establish the most efficient profile ownership structures, educate users with the level of information they need to perform their assigned functions, and prepare guidelines for the various administrators.

Finally, you and the implementation team should prepare an implementation plan to guide the work of the team. Table 5 provides a checklist for the implementation team to use while preparing the implementation plan. Note that this checklist represents only a starting point; it is not meant to be exhaustive.

Table 5. Checklist for Implementation Team Activities

Item	Comments
Objectives	☐ What are the installation's security objectives? ☐ Over what time frame are they to be achieved? ☐ Is management's position clear on all objectives? ☐ Is the statement of security policy clear and complete for all objectives?
Protection	☐ What resource classes are to be protected? ☐ What resources within these classes are to be protected? ☐ Can protection be phased in? ☐ Which resources must be protected, and when?
Naming Conventions	☐ What installation data set or general resource naming conventions already exist? ☐ Are changes necessary? ☐ Does implementing RACF provide an opportunity to enforce naming conventions? ☐ If so, can they be enforced across the entire installation or only over a subset of the installation? ☐ Immediately or eventually?

Table 5. Checklist for Implementation Team Activities (continued)

Item	Comments
Organization	☐ Can the definition of RACF groups (and their associated users) be mapped to the existing organizational structure? ☐ What changes to the organizational structure, if any, are necessary? ☐ How is RACF to be controlled and administered? ☐ Which functions are to be retained centrally? ☐ Which functions are to be delegated, wholly or in part? ☐ Which users should have what RACF attributes?
User and Group Names	☐ What names are to be established for groups and user IDs? ☐ Which groups and users are to be defined to RACF? ☐ Which user verification technique is to be used?
Transparency	☐ Try to make RACF transparent to your users wherever possible. ☐ Which resources can be protected by generic profiles? ☐ Which resources require discrete profiles? ☐ Which users and groups should be placed in the access lists, and with what access authorities? ☐ What deviations from strict user accountability are to be allowed, and for how long?
RACF Tailoring	☐ Which RACF exits are to be used, if any, and under what conditions?
Authorizations	☐ What authorizations are required for the program properties table (PPT), APF libraries, and similar items?
Recovery	☐ What recovery procedures must be established?
Violation Procedures	☐ What security procedures for logging, reporting, and auditing must be established?
Subsystems	☐ What are the security requirements for IMS, CICS, and other subsystems?
Storage Management Subsystem (SMS)	☐ Is your data managed by SMS? ☐ If it is, what is required for your SMS constructs, application IDs, and data set owners?
Test Plan	☐ What is the plan for testing the RACF implementation?
Education	☐ What is the plan for preparing user documentation and other educational material? ☐ Should there be a newsletter for most users and more detailed education for group administrators?
Install RACF	☐ What RACF options are to be used? ☐ What is the plan for installing RACF?
Monitor	☐ After beginning to define groups, users, generic profiles, and data for a pilot group, how will progress against your implementation plan be monitored? ☐ What procedures will be established to ensure that future applications receive the appropriate security considerations?

Chapter 3. Defining Groups and Users

This chapter provides in-depth information on defining groups and users.

Defining RACF Groups

The group structure of RACF can be mapped to the organizational structure that exists at your installation. That is, RACF conforms naturally to a tree structure of groups, where each group except SYS1, which is predefined as the highest group, has a superior, or owning group. Groups can correspond directly to business entities such as divisions, departments, and projects. Users can be connected to one or more groups.

Types of Groups

When you define a group, you should consider the basic purpose of the group. Is it an administrative group, a holding group, a data control group, a functional group, or a user group?

When setting up RACF groups, keep in mind that the maximum number of users that you can connect to any one group is approximately 5900. See the section on determining storage requirements for profiles in *z/OS SecureWay Security Server RACF Macros and Interfaces* for information about how to determine the exact maximum number.

For groups that may become large, and for which a quick listing of members is not needed, you may wish to consider defining the groups using the UNIVERSAL operand of the ADDGROUP command. Universal groups may be appropriate for holding groups and other types of groups. See “Defining Large Groups with the UNIVERSAL Attribute” on page 54.

Administrative Groups

You can create a group simply as an administrative convenience. For example, you might create a group to represent an organizational entity, such as a region or a division.

With RACF delegation, you can create this kind of group for each group administrator. Operating from such groups, the group administrators can then define other groups needed by their local users.

Holding Groups

A popular technique that retains user definition centrally, yet allows the effective use of group administrators, is to establish a *holding group*. You define all users centrally and initially connect them to a group named HOLD with the minimum of authorities. HOLD does not appear in any access lists, and therefore has no real significance to the user.

Group administrators, to whom you give CONNECT (but not JOIN) authority, can connect the appropriate users to the groups under their control and change the users' default group name as appropriate. This technique allows the installation to assign correct account numbers and control other installation considerations while allowing flexibility in the grouping of the user population.

Note: A group cannot contain more than approximately 5900 users. Therefore, if you have more than this number of users, you cannot assign them to a single holding group. Also, you should be aware that extremely large groups can have performance implications for the RACF database. For more information, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Data Control Groups

You can create a group to act as a control point for the protection of data. For example, by using the group SYS1, you can determine which users are permitted to protect the SYS1 data sets. Only users with CREATE authority or higher in this group can protect system data sets. At your location, you or your delegate might consider defining one such group for every high-level qualifier representing data that is to be protected.

For more information, see "Protecting Group Data Sets" on page 153.

Functional Groups

A group can represent a functional area of the installation for the purpose of data sharing. For example, a financial analyst might need to access a variety of resources across many groups, such as accounting, payroll, marketing, and others. Of course, the owners of each resource could permit the financial analyst to access their resources by placing the analyst's user ID on an access list. But if a new financial analyst takes over the job, it is then necessary to add the new user ID to each RACF profile. Likewise, the RACF profiles must be updated when the analyst no longer has a need to access the data. This arrangement involves a great deal of unnecessary activity by the resource owners.

Instead, you can create a group that represents the financial analyst function and permits access to the data defined to the group. Access to the entire range of data can then be managed by controlling the user population in the defined group. For those cases involving one-time access, owners of the needed data would simply PERMIT access by the defined group. Where appropriate, the group name could be included in profile access lists to ensure automatic availability of needed data to the financial analyst group. New financial analysts could be connected to the group, as required, to gain access to the entire range of data. Likewise, analysts could be removed from the group whenever necessary. By controlling the user population of such a functional group, resource profile changes on a day-to-day basis become unnecessary.

User Groups

You can define a group to serve as an anchor point for users who otherwise have no common access requirements. For example, engineers and scientists, as well as other problem-solving users, might have no need to access application-related data in the system. Their only interest might be in their own personal data. You can place this set of users in a single group that has no access to other data.

Also, you can define groups based on access level. For example, if PAY.DATA is a RACF-defined data set, two groups could be defined, PAYREAD and PAYUPDTE, both of which would appear in the PAY.DATA access list, but with READ and UPDATE access, respectively. Any users requiring access would be connected, as appropriate, by the group administrator.

Group Profiles

When you define a group to RACF, you create a group profile in the RACF database. A group profile consists of *segments:* a base segment and optionally, DFP, OMVS, and OVM segments.

Each segment of a group profile consists of *fields*. When you define a group's profile (using the ADDGROUP command) or change a group's profile (using the ALTGROUP command), you can specify the information contained in each field of each segment. To define or change information in a non-base segment of a group profile, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access control.

You can list the contents of an entire group profile or the contents of individual segments of the group profile using the LISTGRP command. To display information in a non-base segment of a group profile, you must have the SPECIAL or AUDITOR attribute or at least READ authority to the segment through field-level access control.

For more information, see "Field-Level Access Checking" on page 213 and "Controlling Access to the DFP Segment" on page 490.

For information on how to use the ADDGROUP, ALTGROUP, and LISTGRP commands, see *z/OS SecureWay Security Server RACF Command Language Reference*.

The Base Segment in Group Profiles

The base segment of a group profile contains basic information that is needed to define a group to RACF. You can specify the following information in the base segment:

groupname Name of the group

OWNER Owner of the group's profile

SUPGROUP Name of the group's superior group

TERMUACC or **NOTERMUACC**

For RACF-protected terminals: Indicates whether to allow access based on the UACC of the terminal profile

DATA Installation-defined data

MODEL Name of the profile used as a model for new group data set profiles, either generic or discrete

UNIVERSAL Universal group

See *z/OS SecureWay Security Server RACF Command Language Reference* for information about the authorization required to create, change, or view information in the base segment.

The DFP Segment in Group Profiles

You can define a DFP segment for group profiles. The DFP segment contains default values that DFP uses to determine data management and DASD storage characteristics for SMS-managed data sets. You can specify the following information in this segment:

DATAAPPL Group's DFP data application identifier

DATACLAS Group's default data class for attributes used during allocation of all new data sets

MGMTCLAS Group's default management class for attributes used in managing a data set after it is allocated

STORCLAS Group's default storage class for logical storage attributes

To define or change information in the DFP segment of a group profile, you must have the SPECIAL attribute or at least UPDATE authority to the segment by way of field-level access control. To display information in the DFP segment of a group profile, you must have the SPECIAL attribute or at least READ authority to the segment by way of field-level access control. For more information, see "Controlling Access to the DFP Segment" on page 490.

The OMVS Segment in Group Profiles

You can use the OMVS segment of the group profile to specify information about the group's z/OS UNIX group. Specifically, when you define a new z/OS UNIX group or change z/OS UNIX attributes for an existing group, you can specify the following information in the group's profile:

GID The group's z/OS UNIX group identifier

To define or change information in the OMVS segment of a group profile, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access control.

To display information in the OMVS segment of a group profile, you must have the SPECIAL attribute or at least READ authority to the segment through field-level access control.

You can authorize users to access to the OMVS segment of a group profile using the GROUP.OMVS.* or the GROUP.OMVS.GID profile in the FIELD class.

When a GID is assigned to a group, all users connected to this group as their default group who have an z/OS UNIX user identifier (UID) in their user profile can use z/OS UNIX functions and can access z/OS UNIX files based on the GID and UID values assigned. If a user's current connect group is not their default group, a GID must also be assigned to the current connect group.

For information, see "Defining Group Identifiers (GIDs)" on page 504 and "Using Default OMVS Segments in USER and GROUP Profiles" on page 506.

The OVM Segment in Group Profiles

You can use the OVM segment of the group profile to specify information about the group's OpenExtensions VM group. When you define a new OpenExtensions VM group or change OVM attributes for an existing group, you can specify the following information in the group's profile:

GID The group's OpenExtensions VM group identifier

To define or change information in the OVM segment of a group profile, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access control.

To display information in the OVM segment of a group profile, you must have the SPECIAL attribute or at least READ authority to the segment through field-level access control.

The TME Segment in Group Profiles

You can define a TME segment for group profiles.You can specify the following information in this segment:

ROLES A list of role profiles that refer to this group

To define or change information in the TME segment of a group profile, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access control.

To display information in the TME segment of a group profile, you must have the SPECIAL attribute or at least READ authority to the segment through field-level access control.

The TME segment fields are intended to be updated only by the Tivoli product, which manages updates, permissions, and cross references. Security administrators should directly update TME fields only on an exception basis.

Defining Large Groups with the UNIVERSAL Attribute

If you are planning to create some groups that may become large and are unlikely to be deleted, such as groups that contain all users, you may define them as universal groups using the UNIVERSAL operand of the ADDGROUP command. *Universal groups* are user groups that do not have complete membership information stored in their group profiles. The benefit is that there is no limit on the number of group members. However, you cannot list all the members using the LISTGRP command.

The UNIVERSAL attribute can be defined for a group only at group creation time. The UNIVERSAL attribute cannot be added or removed using the ALTGROUP command. The output of the LISTGRP command will contain the UNIVERSAL attribute for universal groups.

The group profile of a universal group contains limited group membership information. Only users who have group authorities other than USE, and those who have the group-SPECIAL, group-OPERATIONS or group-AUDITOR attribute, are added to the member list. Therefore, the output of the LISTGRP command will not provide a complete list of group members. The best way to list the members of a universal group is using the output of the database unload utility. See "Using IRRDBU00 with Universal Groups" on page 323. For information about sample RACFICE reports available to assist you, see "Reports Based on the Database Unload Utility (IRRDBU00)" on page 329.

If you wish to delete a universal group, you should use the remove ID utility (IRRRID00) to delete the group and remove all member connections. See “Processing Universal Groups” on page 353 for information about using IRRRID00 to delete universal groups.

Group Naming Conventions

The naming conventions for groups are relatively simple:

- A group name must be 1–8 characters in length, chosen from the letters (A–Z), numbers (0–9), or # (X'7B'), $ (X'5B'), or @ (X'7C'). It may not start with a number.

  Note: These characters may be displayed differently on terminals outside the United States. Therefore, use the characters with the hexadecimal equivalents shown above.
- No two groups can have the same name. No group name can be the same as a user ID.

Because two or more users might want to use the same group name (for example, ADMIN), you should adopt naming standards locally to prevent conflicts. For example, consider assigning a unique 1- or 2-character group name prefix to each group administrator. Then each group defined by a group administrator would have a name that consists of the administrator’s prefix followed by whatever characters the administrator chooses to use. This prefixing ensures that two group administrators cannot use the same group name.

For information about group naming conventions for z/OS UNIX group identifiers (GIDs), see “Defining Group Identifiers (GIDs)” on page 504.

Benefits of Using RACF Groups

Some of the benefits of using RACF groups include:

- Reducing the effort to maintain access lists
- Avoiding the need to refresh in-storage profiles
- Providing a form of timed PERMIT
- Minimizing the length of access lists

  Note: For detailed information, see “Limiting the Size of Your Access Lists” on page 205.

Reducing the Effort of Maintaining Access Lists

Using RACF groups can reduce the time you spend administering access to resources.

Instead of adding and deleting users to the access lists of several profiles, you can create a RACF group and place it on the access list instead of the user IDs. Then, you can give CONNECT group authority in that group to an appropriate person (perhaps the owner of the resource profiles). That person can then change the membership of the group (through the use of the CONNECT and REMOVE commands) instead of issuing the PERMIT command many times to change the access lists of all of the affected resource profiles.

Avoiding the Need to Refresh In-Storage Profiles

If your installation maintains in-storage copies of resource profiles through the SETROPTS RACLIST or SETROPTS GENLIST command, changes to those profiles do not take effect on the system until a SETROPTS RACLIST REFRESH or SETROPTS GENERIC REFRESH command is issued.

To avoid the need to refresh the in-storage copies, place a RACF group on the access list instead of a user ID. Then, give CONNECT group authority to an appropriate person (perhaps the owner of the resource profiles). That person can then change the membership of the group (through the use of the CONNECT and REMOVE commands) instead of issuing the PERMIT command to change the access list of the affected resource profiles, and asking a user with SPECIAL to refresh the in-storage profiles.

Notes:

1. If a user who is already logged onto the system is added to a RACF group with the CONNECT command, that user must log off and log on again before using the group authority to access resources in classes that have been RACLISTed.
2. If a user who is already logged onto the system is deleted from a RACF group with the REMOVE command, that user must log off and log on again before accessing resources in classes that have been RACLISTed without using the group authority.
3. Started tasks must stop and restart to use the new authority. These tasks could include JES2 and JES3.

Providing a Form of Timed PERMIT

You can allow a user to access a protected resource for a limited time by taking the following steps:

1. Ensure that the only access the user has to the resource is by virtue of the fact that the user is connected to a RACF group that has the desired access to the resource. (List the appropriate resource profiles to check for the user's user ID, or other groups to which the user is connected, in the access list. Also, list the user's RACF user profile to check for the OPERATIONS or group-OPERATIONS attribute. Depending on the class of the resource, the OPERATIONS attribute might allow the user to access the resource.)
2. Connect the user to the group with a resume or revoke date. To cause the user's access to stop on a certain date, enter:

   ```
   CONNECT userid GROUP(groupname) REVOKE(date)
   ```

   To cause the user's access to start on a certain date, enter:

   ```
   CONNECT userid GROUP(groupname) RESUME(date)
   ```

> **Attention**
> If the user's membership in the group allows the user to create profiles, and the user becomes the OWNER of such profiles, the user might still have access to the profiles after the revoke date.

Group Ownership and Levels of Group Authority

The following topics discuss group ownership, group authorities, suggestions for assigning group authorities, and the group terminal option.

Ownership of a RACF Group

Each group that you define to RACF must be owned by a RACF-defined user or by its superior group. You assign ownership of a group with the ADDGROUP command when you create a new group profile or with the ALTGROUP command when you change an existing group profile. If you are the owner of a group (or if you are a connected user who has the group-SPECIAL attribute), you have the authority to:

- Define new users to RACF (provided you also have the CLAUTH attribute for the USER class)

- Connect and remove users from the group
- Delegate and change group authorities and set the default UACC for all new resources belonging to members of the group
- Modify, list, and delete the group profile
- Define, delete, and list the names of the subgroups under the group
- Specify the group terminal option

Note: Ownership of a group by a user does not allow that user to update the access lists of resource profiles owned by the group.

For a list of the RACF commands that group owners can issue, see Table 48 on page 581.

Group Ownership of Profiles

You can assign a *RACF group* as the owner of a user profile, group profile, data set profile, or general resource profile. In this way, profile ownership can remain constant, regardless of how often users change jobs in your organization.

Any user connected to the owning group who has the group-SPECIAL attribute has the authority of SPECIAL for all profiles owned by the group (see “User Attributes” on page 73) and also has the ability to perform all owner functions for the group.

You can assign any group to be the owner of a profile. (A group profile must be owned by either a user or its superior group.) An owning group does not need to be a group to which a user (represented by the profile) is connected. Being able to assign any group as an owner gives you flexibility in defining an authority structure. For example, you could establish one group for the sole purpose of owning user profiles, and give a group administrator the group-SPECIAL and CLAUTH (for the USER class) attributes in that group.

Group Authorities

Each user in a group requires a level of group authority for that group. If a user is connected to several groups, the user has a level of group authority for each group. The group authorities are described in Table 6.

Table 6. Group Authorities

Authority	Functions Permitted	RACF Commands Permitted
USE	A user with the USE group authority can enter the system under control of that group, and can access resources (such as data sets, terminals, and others) to which the group is authorized.	LISTDSD, RLIST
CREATE	A user with the CREATE group authority can allocate new group data sets, RACF-protect group data sets, and control access to the profiles he or she has created. However, unless the user has access other than the CREATE group authority itself, the user cannot delete the data sets. CREATE group authority includes the privileges of USE group authority.	ADDSD command for group data set profiles (all operands except NOSET)

Table 6. Group Authorities (continued)

Authority	Functions Permitted	RACF Commands Permitted	
CONNECT	A user with CONNECT group authority can connect users who are already defined to RACF to the group and assign USE, CREATE, or CONNECT group authority to users in the group. CONNECT group authority includes the privileges of both the USE and CREATE group authorities.	All of the above, plus:	
		ALTUSER	GROUP, AUTHORITY, or UACC operands only
		CONNECT	All operands except SPECIAL/NOSPECIAL, OPERATIONS/NOOPERATIONS, and AUDITOR/NOAUDITOR
		LISTGRP	Groupname operand only
		REMOVE	All operands
JOIN	A user with JOIN group authority can define new users and groups to RACF and assign any level of group authority to new users (including the JOIN authority). To define new users, the user with JOIN authority must also have the CLAUTH user attribute for the USER class. When a user defines a new group, it becomes a subgroup of the group in which the user has JOIN authority. JOIN authority includes the privileges of the USE, CREATE, and CONNECT group authorities.	All of the above, plus:	
		ADDGROUP	All operands
		ADDUSER	All operands except OPERATIONS, SPECIAL, and AUDITOR
		ALTGROUP	SUPGROUP operand only (to change the superior group of a group, a user must have JOIN authority in one group and be the owner of or be connected with the group-SPECIAL attribute to another group)
		DELGROUP	All operands
		LISTGRP	Groupname operand only

For a list of the RACF commands that the group authorities allow users to issue, see Table 46 on page 579.

Suggestions for Assigning Group Authorities

As a security or group administrator, you can create different types of administrative structures, according to how you assign group authorities and group ownership. Two examples of possible structures are total and partial delegation.

Total Delegation: With total delegation, one delegate (group owner) is responsible for the administration of a group, the users in the group, and the group resource profiles. In this scheme, the group owner connects to the group with JOIN authority, defines the group resource profiles, and connects other users to the group with USE authority.

Partial Delegation: With partial delegation, you can share the responsibility for the administration of a group, the users in the group, and the group resource profiles. Under this scheme, the owner of the group connects one user to the group with JOIN authority and this user connects other users to the group, giving CREATE authority to one user and USE authority to all other users. In this way, the owner of the group can monitor the group, the user with JOIN authority can monitor the users in the group, and the user with CREATE authority can create and maintain the group's data set profiles.

Another way to share administration responsibilities for the group, the users, and the group's resources is as follows: the owner of the group connects one user to the group with CREATE authority and all other users with USE authority. The owner

of the group can then monitor the group and the users in the group, and the user with CREATE authority can define and control group data sets.

The Group Terminal Option

The group administrator (that is, the owner of a group) can specify a group terminal option for the group by using the ALTGROUP command with the NOTERMUACC operand. With this option, users of the group are authorized to log on to TSO only from those RACF-protected terminals to which they have been specifically authorized access by the PERMIT command. That is, users of the group may not be authorized to log on to TSO from terminals (either RACF-defined or otherwise) based on the universal access authority of the terminals.

Summary of Steps for Defining a RACF Group

This summary presents the steps required by RACF for defining a RACF group. Your installation may require additional steps, depending on your security policy.

1. Prepare to create the group profile as follows:
   - Decide which group is to be the superior group.
   - Decide the group name. (This cannot be the same as a user ID.)
   - Decide who (a user or RACF group) is to be the owner of the new group. (If the group owner is a user, give him or her the information needed to manage the group.)
   - Decide if the group should be a universal group.
   - If your installation is using RACF to protect terminals, and the users in this group are terminal users who are to be limited to specific terminals, consider specifying the NOTERMUACC operand on the ADDGROUP command.
   - If DFSMS is installed, work with the storage administrator to set the initial values in the group's DFP segment.
2. Create the group profile using the ADDGROUP command.

   For example, to create a group for Department A called DEPTA whose owner and superior group is to be a group called ALLDEPT, enter:

   ```
   ADDGROUP DEPTA OWNER(ALLDEPT) SUPGROUP(ALLDEPT)
   ```
3. Connect appropriate users to the new group.
   - Most users should have USE group authority.
   - A few users might need a group authority higher than USE group authority (such as CONNECT).

   For example, to connect department members SUE, LIZ, and GENE to the DEPTA group and also give LIZ and SUE authority to add new users to the group, enter:

   ```
   CONNECT (SUE LIZ) GROUP(DEPTA) OWNER(DEPTA) AUTHORITY(CONNECT)

   CONNECT GENE GROUP(DEPTA) OWNER(DEPTA)
   ```

   These commands assign ownership of each connection to group DEPTA rather than to the issuer of the CONNECT command (the default). Because GENE's authority defaults to USE, GENE can use any of the resources (for example, data sets) that belong to group DEPTA.
4. If the group is to own group data sets, do the following:
   - Create a "top" generic profile for the group data sets in the DATASET class. For example:

     ```
     ADDSD 'DEPTA.**' UACC(NONE)
     ```

- If appropriate, assign the GRPACC user attribute to a member of the group. (However, before assigning the GRPACC user attribute, see Table 18 on page 211.)

5. If the group requires access to RACF-protected resources, give the group the required access using the PERMIT command. For example:

   ```
   PERMIT 'RACF.PROTECT.DATA' ID(DEPTA) ACCESS(READ)
   ```

6. If the group requires access to z/OS UNIX resources, alter the profile to include an OMVS segment with an z/OS UNIX group identifier (GID). For example:

   ```
   ALTGROUP DEPTA OMVS(GID(100))
   ```

Summary of Steps for Deleting Groups

This summary presents the steps required by RACF and related IBM licensed program products to delete groups from RACF. Your installation may require additional steps, depending on your security policy and the products you have installed.

1. Determine if the group is a universal group by using the LISTGRP command, and look for the UNIVERSAL attribute.
2. If the group is not a universal group, use the output of the LISTGRP command to list the members and remove them from the group.

   You can use the REMOVE command to do this. If the user is the owner of any group data set profiles, specify the new owner on the OWNER operand of the REMOVE command. Before removing a user from the user's default connect group, you must first connect the user to a new group (CONNECT command), and then change the user's default connect group to the new group (ALTUSER command).
3. If the group is a universal group, use the remove ID utility (IRRRID00) to remove all members from the group.

   The LISTGRP command may not list all members of a universal group. For information, see "Processing Universal Groups" on page 353
4. Find all data sets associated with this group (that is, the group name is the high-level qualifier of the data set name) and take the following steps:
   a. Delete or rename (with a new high-level qualifier) the group's group data sets. If you rename or delete a data set that is protected by a discrete profile, the discrete profile is also renamed or deleted.

      Note: You can do this using the DATA SET LIST utility of ISPF.
   b. Identify all of the remaining (generic) data set profiles, create new ones modeled on them if needed, and delete the remaining profiles.

      Attention
      Make sure that you do not delete an old profile unless it is no longer needed.

      Notes:

      1) You can use the following SEARCH command to identify these profiles:

         ```
         SEARCH MASK(groupname.) CLIST('LISTDSD ' ' ALL')
         ```

         As specified, the CLIST operand generates a CLIST that you can run to list all of the information in the data set profiles. This can help you assess whether to use the profiles as models.

2) You can use the FROM operand on the ADDSD command to create new profiles modeled on the old profiles.

5. To research the following steps, use the IRRRID00 utility to list the occurrences of the group name in the RACF database. For information, see “Using the RACF Remove ID Utility (IRRRID00)” on page 342.
6. For each subgroup of the group to be deleted, change the subgroup's superior group to an existing group.

   ```
   ALTGROUP subgroupname SUPGROUP(new-superior-groupname)
   ```
7. If the group is the owner of any profiles (the group's group name was specified on the OWNER operand), change the owner of the profiles to a new group or user.

   Note: To do this, use the appropriate command for changing profiles, such as ALTUSER, ALTGRP, ALTDSD, or RALTER.
8. Remove the group from any access lists in which the group's group ID is specified.

   Note: To do this, use the DELETE operand on the PERMIT command.
9. After all occurrences of the group name are deleted from the RACF database, use the DELGROUP command to delete the group profile.

Defining Users

As a general objective, all users should be defined to RACF. Users who are not defined to RACF can use the system virtually unimpeded, unless, of course, they attempt to access data to which they are unauthorized.

The users you must initially define are those you have selected for the pilot project and the central core of personnel who maintain and operate the system itself. Other users can then be defined as determined by convenience and the priority of their security needs.

You should consider defining the following users to RACF:

- Interactive users of CICS, IMS, TSO/E, NetView, or other products that support logging on at a terminal.
- z/OS UNIX users. You use RACF commands to define users to z/OS UNIX. The z/OS UNIX attributes are kept in the OMVS segment of the user's profile and can be specified in addition to any existing attributes. The new attributes extend the user's capabilities to include the use of z/OS UNIX functions. In order to use z/OS UNIX services, a user must have z/OS UNIX attributes defined, such as an z/OS UNIX user identifier (UID) in his or her user profile and an z/OS UNIX group identifier (GID) in the group profile of his or her current connect group (the user's default group or the one specified on the TSO LOGON screen or job card). For more information, see “Chapter 18. RACF and z/OS UNIX” on page 503.
- Users who submit batch jobs without first logging on to a terminal (such as through a physical card reader).
- MVS or JES system operators. You should work with your MVS or JES system programmer to determine which MVS and JES system operators should be defined to RACF. For more information, see “Defining and Grouping Operators” on page 440.
- Started procedures.
- Node names in an NJE network.

- RJP or RJE remote workstations or nodes.
- Console IDs if LOGON(AUTO) is specified in the CONSOL*xx* member of SYS1.PARMLIB. For more information, see *z/OS MVS Initialization and Tuning Reference*.

There are some advantages in defining *all* users to RACF:

- Defining all users provides for better administrative control over who is using the system. This in turn can reduce misuse of system resources.
- Attempted violations by undefined users are difficult to investigate, because they do not have user IDs that are associated with real persons or processes.

Whether all users are eventually defined to RACF is your decision. You may deem individual accountability for a certain section of the user population unnecessary in some cases. Note that this can reduce your ability to determine exactly who took security-relevant actions.

User Profiles

When you define a user to RACF, you create a user profile in the RACF database. A user profile consists of a base segment and, optionally, any of the following segments: CICS, DCE, DFP, LANGUAGE, LNOTES, NDS, NETVIEW, OMVS, OPERPARM, OVM, TSO, and WORKATTR.

Each segment of a user profile consists of fields. When you define a user's profile (using the ADDUSER command) or change a user's profile (using the ALTUSER command), you can specify the information contained in each field of each segment of the profile.

To define or change information in a non-base segment of a user profile, including your own, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access checking.

To list the contents of a user profile or the contents of individual segments of the user profile, use the LISTUSER command.

To display the information in a non-base segment of a user profile, including your own, you must have the SPECIAL or AUDITOR attribute or at least READ authority to the segment through field-level access checking.

IBM recommends that you use field-level access control to let users view, and optionally modify, some or all of the information in the non-base segments of their user profiles.

For more information, see "Field-Level Access Checking" on page 213, "Controlling Access to the DFP Segment" on page 490, and "Field-Level Access Checking for TSO" on page 499.

When you use the RACDCERT command to add a certificate definition and associate it with a specified RACF-defined user ID, information about the definition is added to the user profile. To see the certificate definitions, enter:

```
RACDCERT LIST
```

To issue this command, you must have one of the following authorities:

- The SPECIAL attribute
- Sufficient authority to resource IRR.DIGTCERT.LIST in the FACILITY class:

  - READ access to IRR.DIGTCERT.LIST to list this information for yourself
  - UPDATE access to IRR.DIGTCERT.LIST to list this information for others

When you use the RACDCERT command to add a certificate name filter and associate it with a specified RACF-defined user ID, information about the definition is added to the user profile. To see the certificate name filter definitions, enter:

```
RACDCERT LISTMAP
```

To issue this command, you must have one of the following authorities:

- The SPECIAL attribute
- Sufficient authority to resource IRR.DIGTCERT.LISTMAP in the FACILITY class:
  - READ access to IRR.DIGTCERT.LISTMAP to list this information for yourself
  - UPDATE access to IRR.DIGTCERT.LISTMAP to list this information for others

When you use the RACDCERT command to add a certificate key ring and associate it with a specified RACF-defined user ID, information about the definition is added to the user profile. To see the ring definitions, enter:

```
RACDCERT LISTRING
```

To issue this command, you must have one of the following authorities:

- The SPECIAL attribute
- Sufficient authority to resource IRR.DIGTCERT.LISTRING in the FACILITY class:
  - READ access to IRR.DIGTCERT.LISTRING to list this information for yourself
  - UPDATE access to IRR.DIGTCERT.LISTRING to list this information for others

When you use the RACLINK command to establish a user ID association, information about the association is added to the user profile. To see the user ID associations, enter:

```
RACLINK LIST
```

For more information on how to use the ADDUSER, ALTUSER, LISTUSER, RACDCERT, and RACLINK commands, see *z/OS SecureWay Security Server RACF Command Language Reference*.

The Base Segment in User Profiles

The base segment of a user profile contains basic information that is needed to define a user to RACF. You can specify the following information in the base segment:

USERID User's identification

NAME User's name

OWNER Owner of the user's profile

DFLTGRP User's default group

AUTHORITY User's authority in the default group

PASSWORD User's password

NOPASSWORD
Gives the user the PROTECTED attribute when the user has the NOOIDCARD attribute

REVOKE Date on which RACF prevents the user from having access to the system

RESUME Date on which RACF lets the user have access to the system again

UACC Default universal access authority for resources that the user defines

WHEN Days of the week and hours of the day during which the user has access to the system

ADDCATEGORY
User's installation-defined security category

SECLEVEL User's installation-defined security level

CLAUTH Classes in which the user can define profiles

SPECIAL Gives the user the system-wide SPECIAL attribute

AUDITOR Gives the user the system-wide AUDITOR attribute

OPERATIONS Gives the user the system-wide OPERATIONS attribute

DATA Installation-defined data

ADSP Indicates that all permanent data sets the user creates are to be RACF-protected with discrete profiles

GRPACC Indicates that other group members can have access to any group data set the user protects with a data set profile

MODEL Name of the data set model profile to be used when creating new data set profiles, either generic or discrete

OIDCARD Indicates that the user must supply an operation ID card when logging on to the system

RESTRICTED Indicates that global access checking, the `ID(*)` entry on the access list, and the UACC will not be used to allow this user access to a protected resource

SECLABEL User's default security label

CERTNAME The names of the profiles in the DIGTCERT class that are associated with this RACF user ID

CERTLABL The certificate labels for the profiles in the DIGTCERT class that are associated with this RACF user ID

CERTPUBK The public key associated with a public key certificate. This is the BER-encoded public key as specified in the certificate.

CERTSJDN The subject name of the entity to whom the certificate is issued. This is the BER-encoded format of the subject's distinguished name as contained in the certificate.

Note: You can only add or delete the data in the CERTNAME, CERTLABL, CERTPUBK and CERTSJDN fields by using the RACDCERT command. The ADDUSER or ALTUSER commands have no effect on these fields.

NMAPNAME The names of the profiles in the DIGTNMAP class containing certificate name filters that are associated with this RACF user ID

NMAPLABL The labels for the certificate name filters that are associated with this RACF user ID

See *z/OS SecureWay Security Server RACF Command Language Reference* for information about the authorization required to create, change, or view information in the base segment.

The CICS Segment in User Profiles

You can specify information for CICS terminal operators in RACF user profiles. If CICS/ESA 3.2 or later is installed, this information is used when CICS terminal operators sign on to CICS.

Note: Before you define or change CICS segment information, make sure that CICS 3.2 or later is installed on your system.

For planning information, see *CICS RACF Security Guide*.

An installation can set the default characteristics (including authorities) for CICS terminal operators by defining CICS segments in the user profiles of these users. To do this, issue the ADDUSER or ALTUSER command with the CICS operand. You can specify the following information:

OPCLASS Classes assigned to this operator to which basic mapping support (BMS) messages are to be routed

OPIDENT Identification of the operator for use by BMS

OPPRTY Priority of the operator

TIMEOUT Time that the operator is allowed to be idle before being signed off

XRFSOFF Indicates whether the operator is to be signed off by CICS when an XRF takeover occurs

To define or change information in the CICS segment of a user profile, including your own, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access checking. To display information in the CICS segment of a user profile, you must have the SPECIAL attribute or at least READ authority to the segment through field-level access checking. For more information, see “Field-Level Access Checking” on page 213.

The DCE Segment in User Profiles

The DCE segment, defined to the RACF user profile, associates a DCE principal with the RACF user profile.

You can specify the following attributes:

DCENAME User’s DCE principal name

UUID User’s DCE principal universal unique identifier

HOMECELL Home cell for this DCE user

HOMEUUID Home cell universal unique identifier

AUTOLOGIN Single signon processing (YES or NO)

As RACF administrator, you need to work with the DCE administrator to define RACF profiles to use these features correctly.

The DFP Segment in User Profiles

You can define a DFP segment for user profiles. The DFP segment contains default values that DFP uses to determine data management and DASD storage characteristics for user data sets.

You can specify the following information in this segment:

DATAAPPL User's DFP data application identifier

DATACLAS User's default data class for attributes used during allocation of all new data sets

MGMTCLAS User's default management class for attributes used in managing a data set after it is allocated

STORCLAS User's default storage class for logical storage attributes

To define or change information in the DFP segment of a user profile, including your own, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access checking. To display information in the DFP segment of a user profile, you must have the SPECIAL attribute, the AUDITOR attribute, or at least READ authority to the segment through field-level access checking. For more information, see "Controlling Access to the DFP Segment" on page 490.

The KERB Segment in User Profiles

You can define a KERB segment for user profiles. The KERB segment contains the information about a Network Authentication Service user, such as the local principal name that will be mapped to this RACF user ID.

You can specify the following information in this segment:

KERBNAME User's local principal name

MAXTKTLFE User's maximum ticket life

ENCRYPT User's key encryption types

To define or change information in the KERB segment of a user profile, including your own, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access checking. To display information in the KERB segment of a user profile, you must have the SPECIAL attribute, the AUDITOR attribute, or at least READ authority to the segment through field-level access checking. For more information, see "Defining Local Principals" on page 557.

The LANGUAGE Segment in User Profiles

You can specify a user's preferred national languages in the LANGUAGE segment of the user's user profile. The languages stored in the user profiles are used by TSO, CICS/ESA 3.2 or later, and any other applications that use the RACROUTE REQUEST=EXTRACT macro to determine a user's preferred national languages. For specific information on how an application checks for a user's preferred national languages, see the documentation for that product.

Note: In general, you should also specify an installation default using the LANGUAGE operand of the SETROPTS command.

If individual users prefer languages other than the installation defaults, use the LANGUAGE operand of the ADDUSER or ALTUSER command to assign the preferred languages. On the LANGUAGE operand, you can specify the following information:

PRIMARY User's preferred national language, if different from the installation default

SECONDARY User's alternate national language, if different from the installation default

To define or change information in the LANGUAGE segment of a user profile, including your own, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access checking. To display information in the LANGUAGE segment of a user profile, you must have the SPECIAL attribute or at least READ authority to the segment through field-level access checking. For more information, see “Field-Level Access Checking” on page 213.

The LNOTES Segment in User Profiles

You can define an LNOTES segment for user profiles. The LNOTES segment contains the Lotus Notes for z/OS short name that will be mapped to this RACF user ID.

You can specify the following information in this segment:

SNAME User’s short name for use with Lotus Notes for z/OS

To define or change information in the LNOTES segment of a user profile, including your own, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access checking. To display information in the LNOTES segment of a user profile, you must have the SPECIAL attribute, the AUDITOR attribute, or at least READ authority to the segment through field-level access checking. For more information, see “RACF Support for NDS and Lotus Notes for z/OS” on page 295.

The NDS Segment in User Profiles

You can define an NDS segment for user profiles. The NDS segment contains the Novell Directory Services for OS/390 user name that will be mapped to this RACF user ID.

You can specify the following information in this segment:

UNAME User’s user name for use with Novell Directory Services for OS/390

To define or change information in the NDS segment of a user profile, including your own, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access checking. To display information in the NDS segment of a user profile, you must have the SPECIAL attribute, the AUDITOR attribute, or at least READ authority to the segment through field-level access checking. For more information, see “RACF Support for NDS and Lotus Notes for z/OS” on page 295.

The NETVIEW Segment in User Profiles

When you define a new NetView operator or change NETVIEW attributes for an existing operator, you can specify the following information in the NETVIEW segment of the user’s profile:

CONSNAME MCS console identifier

CTL Specifies GLOBAL, GENERAL, or SPECIFIC control

DOMAINS Domain identifier

IC Initial command or list of commands to be executed by NetView when this NetView operator logs on

MSGRECVR Indicates whether the operator will receive unsolicited messages

NGMFADMN Indicates whether this operator can use the NetView graphic monitor facility

OPCLASS Class of the operator

The OMVS Segment in User Profiles

When you define a new z/OS UNIX user or change z/OS UNIX attributes for an existing user, you can specify the following information in the OMVS segment of the user's profile:

HOME User's z/OS UNIX initial directory path name

PROGRAM User's z/OS UNIX program path name, such as a default shell program

UID User's z/OS UNIX user identifier

CPUTIMEMAX
User's z/OS UNIX `RLIMIT_CPU` (maximum CPU time)

ASSIZEMAX User's z/OS UNIX `RLIMIT_AS` (maximum address space size)

FILEPROCMAX
User's z/OS UNIX maximum number of files per process

PROCUSERMAX
User's z/OS UNIX maximum number of processes per UID

THREADSMAX
User's z/OS UNIX maximum number of threads per process

MMAPAREAMAX
User's z/OS UNIX maximum memory map size

To define or change information in the OMVS segment of a user profile, including your own, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access checking. To allow authorization to the entire OMVS segment of a user profile, the user would need authority to the USER.OMVS.* profile in the FIELD class. Individual fields in the OMVS segment can be defined such as USER.OMVS.UID. You can allow users to change their own HOME or PROGRAM values by creating USER.OMVS.HOME and USER.OMVS.PROGRAM in the FIELD class and permitting &RACUID to the profiles.

For more information, see "Defining User Identifiers (UIDs)" on page 504 and "Using Default OMVS Segments in USER and GROUP Profiles" on page 506.

The OPERPARM Segment in User Profiles

Users can enter an extended MCS console session as follows:

- If the application they are running issues the MCSOPER macro (which is an authorized macro)
- If the user issues the TSO CONSOLE command

For information on using RACF to control the users who can establish an extended MCS console session, see *z/OS MVS Planning: Operations.*

Users who enter an extended MCS console session can act as system operators. An installation can set the default characteristics (including command authorities) for these sessions by defining OPERPARM segments in the user profiles of these users. To do this, issue the ADDUSER or ALTUSER command with the OPERPARM operand. You can specify the following information:

ALTGRP Alternative console group for recovery

AUTH Operator's command authority

ROUTCODE Routing codes that the operator receives

LEVEL Message level that the operator receives

MFORM Format in which messages are displayed

MSCOPE Name of the system from which the operator receives unsolicited messages

CMDSYS Name of the system to which the operator is connected for command processing

MONITOR Events that the operator can monitor

LOGCMDRESP
Indicates whether command responses received by the operator are to be recorded on the hardcopy log

MIGID Indicates whether the operator is to receive a migration console ID

STORAGE Maximum amount of virtual storage in megabytes for message queuing

DOM Indicates whether the operator should receive delete operator message requests

KEY Indicates a data retrieval key used to search for operator consoles using the DISPLAY CONSOLES command

UD Indicates whether the operator should receive messages that are considered undeliverable

To define or change information in the OPERPARM segment of a user profile, including your own, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access checking. To display information in the OPERPARM segment of a user profile, you must have the SPECIAL attribute or at least READ authority to the segment through field-level access checking. For more information, see “Field-Level Access Checking” on page 213.

The OVM Segment in User Profiles

When you define a new OpenExtensions VM user or change OVM attributes for an existing user, you can specify the following information in the OVM segment of a user’s profile:

UID The user’s OpenExtensions VM user identifier

HOME The user’s OpenExtensions VM initial directory path name

PROGRAM The user’s OpenExtensions VM program path name, such as a default shell program

FSROOT The user’s OpenExtensions VM file system root directory

To define or change information in the OVM segment of a user profile, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access control.

To display information in the OVM segment of a user profile, you must have the SPECIAL attribute or at least READ authority to the segment through field-level access control.

The TSO Segment in User Profiles

When you define a new TSO user or change TSO attributes for an existing user, you can specify the following information in the TSO segment of a user’s profile:

ACCTNUM User’s default account number

COMMAND Command to be run during TSO/E logon

JOBCLASS Default value for user's job class

MSGCLASS Default value for the user's message class

HOLDCLASS Default value for the user's hold class

SYSOUTCLASS
Destination ID for the user's SYSOUT data sets

PROC User's default logon procedure

MAXSIZE User's maximum region size

SIZE User's default region size

SECLABEL Security label specified when the user previously logged on to TSO

UNIT Default device used for allocations

USERDATA Optional user data

If a user logs on to TSO and you have defined a TSO segment in the user's profile, TSO checks the user's authority to use certain TSO resources such as account numbers and logon procedures. If the user is authorized to use a resource such as an account number, TSO continues building a session for the user. Otherwise, TSO prompts the user for a valid account number.

If a user logs on to TSO and you have not defined a TSO segment for that user, TSO checks the SYS1.UADS data set for the information it needs to build a session. If TSO does not find an entry for the user in SYS1.UADS, the user is denied access to the system.

You can move TSO user attribute information from SYS1.UADS to the RACF database. (SYS1.UADS contains an entry for each TSO user that describes the attributes that regulate the user's access to the system.) When you move this TSO information into the RACF database, it is stored in the TSO segment of the user's profile. When a user logs on to TSO, it uses the information contained in the TSO segment to build a session for the user.

Moving the TSO user information to the RACF database eliminates the need to maintain an entry in SYS1.UADS for each TSO user. However, you *must* maintain entries in SYS1.UADS for certain users, such as IBMUSER and system programmers. For example, if you need to deactivate RACF to perform maintenance on the RACF database, users authorized to perform this maintenance must be able to log on to the system. When RACF is inactive, TSO checks entries in SYS1.UADS to authorize access to the system.

Notes:

1. You can use the RACONVRT EXEC to help convert SYS1.UADS entries to RACF user profiles. See *z/OS TSO/E Customization* for more information.
2. If you are defining TSO segments in user profiles, you must activate the following TSO general resource classes: TSOPROC and ACCTNUM. For more information, see "Protecting TSO Resources" on page 496.
3. IBM recommends that you use field-level access control to protect fields within the TSO segment of user profiles. Otherwise, any user can list and change the information contained in this segment. For more information, see "Field-Level Access Checking" on page 213.

4. A TSO user can use the TSO/E logon panel to specify or override certain information in the TSO segment of his or her user profile. For example, a user can change an account number, or specify an account number if one has not been specified, using the TSO/E logon panel. RACF checks the user's authorization to the ACCTNUM profile that protects the specified account number. If the user is authorized to use the specified account number, TSO stores the account number in the TSO segment of the user's profile and uses it as a default value the next time the user logs on to TSO. Otherwise, RACF denies access to the account number.

   If users attempt to change their user profiles when logging on, the logon is allowed but the TSO segment is not updated in either of the following cases:
   - The RACF database is *locked*.
   - The system is enabled for sysplex communication and RACF is in read-only mode.

   You must apply APAR OY56802 to TSO/E 2.3, TSO/E 2.3.1 or TSO/E 2.4 to update the TSO segment. See *z/OS SecureWay Security Server RACF System Programmer's Guide* for information on utilities that *lock* the RACF database.

   See *z/OS TSO/E User's Guide* for a description of the information that a user can specify on the TSO/E logon panel.
5. A TSO installation can write a TSO logon pre-prompt exit to bypass checking SYS1.UADS for user attribute information. See *z/OS TSO/E Customization* for more information.

The WORKATTR Segment in User Profiles

You can specify work attribute (WORKATTR) segments in user profiles that are set up to support APPC requests running under an APPC transaction program. WORKATTR segments include SYSOUT and account information for the users.

An installation can set the default characteristics (including authorities) for these users by defining WORKATTR segments in the user profiles of these users. To do this, issue the ADDUSER or ALTUSER command with the WORKATTR operand.

You can specify the following information in the WORKATTR segment of a user's profile:

WANAME User name on SYSOUT

WABLDG Building on SYSOUT

WADEPT Department on SYSOUT

WAROOM Room on SYSOUT

WAADDR1 SYSOUT address line 1

WAADDR2 SYSOUT address line 2

WAADDR3 SYSOUT address line 3

WAADDR4 SYSOUT address line 4

WAACCNT Account number

To define or change information in the WORKATTR segment of a user profile, including your own, you must have the SPECIAL attribute or at least UPDATE authority to the segment through field-level access checking. To display information in the WORKATTR segment of a user profile, you must have the SPECIAL attribute

or at least READ authority to the segment through field-level access checking. For more information, see “Field-Level Access Checking” on page 213.

User Naming Conventions

The rules for naming users, like those for naming groups, are simple:

- A RACF user ID must be from 1 to 8 characters in length, and may consist of any combination of A-Z, 0-9, # (X'7B'), $ (X'5B'), or @ (X'7C').

  Note: Although RACF permits 8-character user IDs, keep in mind that TSO user IDs and user IDs on JOB cards cannot be more than 7 characters. TSO and MVS also require that the first character of user IDs be A-Z, # (X'7B'), $ (X'5B'), or @ (X'7C').
- The #, $, and @ characters may be displayed differently on terminals outside the United States; therefore, use the characters with the hexadecimal equivalents shown above.
- No two user IDs can be the same. No user ID can be the same as a group name.

For information about user naming conventions for z/OS UNIX user identifiers (UIDs), see “Defining User Identifiers (UIDs)” on page 504.

Suggestions for Defining User IDs

Basically, there are no requirements for establishing a specific type of user ID. That is, in some installations, you might form user IDs by adding a numerical suffix to a group name (for example, ADMIN01 or MKT06). In other cases, you might use first names (for example, PETER and PAUL could be defined and connected to the RESEARCH group. In this case, if PETER subsequently leaves the RESEARCH group to join the TEST group, he need not change his user ID.)

The concept of user IDs based on group names appears practical because a quick glance at the user ID reveals the group. However, this concept might not prove so practical a few years later if many of the current users have changed groups. In addition, how does such a user handle the “userid.data.sets” after the user ID is changed? In the long run, user IDs based on something like user names or personnel numbers do not have this problem and offer the greatest long-term flexibility.

For suggestions related to z/OS UNIX, see “Defining User Identifiers (UIDs)” on page 504.

Migrating Existing User IDs to RACF

Where user IDs already exist in machine readable form (for example, in SYS1.UADS), a simple CLIST can provide a valuable administrative aid in migrating users to RACF.

Note: You can use the RACONVRT EXEC to help convert SYS1.UADS entries to RACF user profiles. For more information, see *z/OS TSO/E Customization*.

Creating New User IDs from Scratch

Where user IDs are assigned from scratch, they can often be created in blocks, using a CLIST. For example, you could centrally create 50 user IDs, MKT01 through MKT50, and allocate them to the manager of group MKT to assign to users in the department. The default group (MKT), password, and other operands can all be preset. You should assign the REVOKE attribute to unused user IDs.

Creating User IDs for System Operators

You can create user profiles for system operators with the ADDUSER command. An operator who already has a TSO user ID, for example, could use this user ID to log on to a console without the need for creating a new user ID.

Note: To keep new passwords from showing up in the system log, system operators logging on to a console should change their passwords only when they log on to the system.

Creating User IDs for RRSF Users

Some users might need to use RRSF functions, including:

- Command direction
- Password synchronization
- Use of the RACLINK command to establish and approve user ID associations.

See “Chapter 10. The RACF Remote Sharing Facility (RRSF)” on page 355 for more information.

Ownership of a RACF User Profile

Each user defined to RACF has a user profile and all user profiles have another RACF user or group as the owner. The owner (or a user who is connected to the owning group and has the group-SPECIAL attribute, or someone with SPECIAL) can change, list, and delete the user’s profile and also has control over the user’s attributes (including the ability to prevent the user from entering the system).

For a list of the RACF commands that owners of user profiles can issue, see Table 48 on page 581.

User Attributes

User attributes are extraordinary capabilities, limitations, or environments that can be assigned to a user either all of the time or when the user is connected to a specific group or groups. When an attribute is to apply all of the time, it is specified at the system level and is called a user attribute. When an attribute is to apply only to a specified group or groups, it is specified at the group level and is called a group-related user attribute. For example, user attributes that you specify in an ADDUSER or ALTUSER command are stored in the user’s profile and are in effect regardless of the group to which the user is connected.

The user attributes are:

SPECIAL
AUDITOR
OPERATIONS
CLAUTH
REVOKE
GRPACC
ADSP
RESTRICTED

The SPECIAL Attribute

A user who has the SPECIAL attribute can issue all RACF commands. The SPECIAL attribute gives the user full control over all of the RACF profiles in the RACF database.

The SPECIAL attribute can be delegated only by a user who has the SPECIAL attribute. It should be limited to the RACF security and group administrators.

Persons who have the SPECIAL attribute should be required to use operator identification cards and passwords, and should change their passwords often to help ensure password security.

Note: Because any user can access an unprotected resource, users who have the SPECIAL attribute should take care to protect their own data sets, because they can contain sensitive information.

You can assign the SPECIAL attribute at the group level. When you do, the *group-SPECIAL* user has full control over all of the profiles within the scope of the group. For additional details, see “User Attributes at the Group Level” on page 79.

For a list of the RACF commands that this attribute allows users to issue, see Table 42 on page 577.

The AUDITOR Attribute

A user who has the AUDITOR attribute has the authority to specify logging options on the ALTDSD, ALTUSER, RALTER, and SETROPTS commands. In addition, the auditor can list auditing information using the LISTDSD, RLIST, LISTUSER, LISTGRP, and SEARCH commands, as well as the IRRUT100 utility. The AUDITOR attribute gives the auditor control of logging to the SMF data set. Logging to SMF helps to detect changes (or attempted changes) to the RACF database and accesses (or attempted accesses) of RACF-protected resources.

The user who has the AUDITOR attribute can list all of the profile information that is available to the SPECIAL user, as well as information that is available to auditors. Note, however, that this extended listing authority does not give the auditor additional access to protected data or additional authority to change information in the RACF database.

If the DSMON program (ICHDSM00) is not defined in the PROGRAM class (it is not a controlled program), a user must have the AUDITOR attribute to run the DSMON program. (If DSMON is a controlled program, the AUDITOR attribute is not enough to run it. The user, or the user’s group, must be in the access list of the DSMON profile, ICHDSM00, to run the DSMON program.)

You should assign the AUDITOR attribute only to users who are responsible for auditing RACF security controls and functions. To provide a check and balance on RACF security measures, you should give the AUDITOR attribute to security or group administrators other than those who have the SPECIAL attribute.

The AUDITOR attribute can be assigned only by a user (security or group administrator) who has the SPECIAL attribute.

Note: Because any user can access an unprotected resource, users who have the AUDITOR attribute should take special care to protect their own data sets, because they can contain sensitive information.

You can assign the AUDITOR attribute at the group level. When you do, the *group-AUDITOR* user’s authority is limited to profiles that are within the scope of that group. For detailed information, see “User Attributes at the Group Level” on page 79.

For a list of the RACF commands that this attribute allows users to issue, see Table 43 on page 578.

The OPERATIONS Attribute

A user who has the OPERATIONS attribute has full access authorization to all RACF-protected resources in the DATASET, DASDVOL, GDASDVOL, PSFMPL, TAPEVOL, VMBATCH, VMCMD, VMMDISK, VMNODE, and VMRDR classes, with the following exceptions:

- If users, their current connect group, or any of their connect groups (if list-of-groups checking is active) is in the access list of a resource profile, they have only the access specified in the access list. For this reason, you should plan carefully before making users who have the OPERATIONS attribute members of any group that is in the access lists of resource profiles.
- Security classification checking or security label checking can deny access.

In addition to having access authorization, an OPERATIONS user can:

- Copy, reorganize, catalog, and scratch user or group data sets.

  Note: The OPERATIONS attribute is required if you use Data Facility Data Set Services (DFDSS) to copy data sets that result in a DEFINE or a discrete data set profile for data sets you do not own.
- Perform input/output operations on tape volumes.
- Create or destroy labels on tape volumes through OPEN and end-of-volume operations.
- Create group data sets for groups.

  An OPERATIONS user *cannot* create group data sets for groups when *both* of the following are true:

  1. The user is connected to the group with less than CREATE authority
  2. The user has less than ALTER access to the data set if it is protected by a generic profile

  If the user has the group-OPERATIONS attribute (that is, the user is connected to a superior group with the OPERATIONS attribute), the group for which the new data set is being created must be within the scope of that superior group.
- Create user data sets. If the user has the group-OPERATIONS attribute (that is, the user is connected to a group with the OPERATIONS attribute), the high-level qualifier of the new data set must be the ID of a user who is within the scope of that group.

  In addition, RACF creates a discrete profile for the user data set if the OPERATIONS user does one of the following:

  – Has the automatic data set protection (ADSP) attribute
  – Specifies PROTECT on the TSO ALLOCATE command that creates the data set
  – Specifies PROTECT=YES or SECMODEL=*profile-name* on the JCL DD statement that creates the data set
- Define profiles for group data sets when one of the following is true:
  – The user is *not* connected to the group of the new data set. If the user has the group-OPERATIONS attribute (that is, the user is connected to a superior group with the OPERATIONS attribute), the group for which the new data set is being created must be within the scope of that superior group.
  – The user is connected to the group with at least CREATE group authority.

Limiting the Capabilities of the OPERATIONS Attribute: You can limit the access to existing resources allowed by the OPERATIONS attribute in two ways:

- By placing the OPERATIONS user (or a group to which the user is connected) in the access list of sensitive resources (using the PERMIT command). The specific access authority (such as NONE or READ) takes precedence over the OPERATIONS attribute.
- By using security levels, security categories, or security labels

You can limit the ability of the OPERATIONS user to create group data sets by ensuring that both of the following are true:

1. The user is connected to the group with less than CREATE authority
2. The user has less than ALTER access to the data set if it is protected by a generic profile

Because the OPERATIONS attribute permits wide access to resources, you should assign this attribute to a minimum number of people. You should also consider auditing those users to whom you have assigned the OPERATIONS attribute. To do this, a user with the AUDITOR attribute must issue the following command:

```
SETROPTS OPERAUDIT
```

To reduce the number of users who have the OPERATIONS attribute at the system level (and therefore have the attribute for all resources in the system), you can assign the OPERATIONS attribute at the group level. When you do, the *group-OPERATIONS* user's authority is limited to resources within the scope of the group. For more information, see "The Scope of Authority for the Users with Group-Level Attributes" on page 79 and "User Attributes at the Group Level" on page 79.

OPERATIONS and DASDVOL Authority: If a person needs to perform maintenance activities on DASD volumes, it is more efficient (for RACF processing) and better (for limiting the resources the person can access) to give the person authority to those volumes using the PERMIT command than to assign the person the OPERATIONS or group-OPERATIONS attribute. To give the person authority to those DASD volumes, define the volumes to RACF and add the person to the access list with the access authority required by the particular resource manager (such as DFDSS). For more information, see "DASD Volume Authority" on page 173.

If you have DFDSS or DFSMS, another alternative is to designate the user as a DFDSS-authorized storage administrator. This method has certain advantages over both OPERATIONS and DASDVOL authorization. For more information, see "DFDSS-Authorized Storage Administration" on page 174.

The OPERATIONS attribute can be delegated only by a user (security or group administrator) who has the SPECIAL attribute.

For a list of the RACF commands that the OPERATIONS attribute allows users to issue, see Table 44 on page 578.

The CLAUTH (Class Authority) Attribute

Users receive the CLAUTH attribute on a *class-by-class* basis. You cannot assign the CLAUTH attribute at the user or group level. If a user has the CLAUTH attribute in a class, or in a class that shares the same POSIT value in the class descriptor table (CDT), RACF allows the user to define profiles in that class.

The classes you can specify with CLAUTH are the USER class and any general resource class.

Notes:

1. The authority of all users to define profiles in general resource classes can be limited by issuing the SETROPTS GENERICOWNER command. For more information, see “Restricting the Creation of General Resource Profiles (GENERICOWNER Option)” on page 117.
2. You must activate the class for which a user has the CLAUTH attribute to enable the user to define profiles in that class.
3. A user’s authority to define profiles extends to any class that has the same POSIT value in the class descriptor table (CDT). For example, if you give a user CLAUTH(TERMINAL), that user can also define profiles in class GTERMINL, because both of these classes have the same POSIT value. For the POSIT values of the classes supplied by IBM, see the description of the class descriptor table (CDT) in *z/OS SecureWay Security Server RACF Macros and Interfaces*.

   You should give the CLAUTH attribute only to those users who are responsible for defining profiles to RACF in the specified classes and in any classes with the same POSIT value.
4. A user to whom you assign the CLAUTH attribute for the USER class is authorized to define new users to RACF with the ADDUSER command, as long as the user is the owner of or has JOIN authority in the new user’s default group.

The CLAUTH attribute can be delegated only by a user with the SPECIAL attribute, or by a user who has both the authority to update the user profile and the CLAUTH attribute for the class authority being delegated.

For a list of the RACF commands that the CLAUTH attribute allows users to issue, see Table 45 on page 578.

The REVOKE Attribute

You can prevent a RACF user from entering the system by assigning the REVOKE attribute on the ALTUSER command. This attribute is useful when you want to prevent a user from entering the system but you cannot use the DELUSER command because the user still owns RACF resource profiles.

You can also assign the REVOKE attribute on a group level by using the CONNECT command. If the user has the REVOKE attribute for a group, the user cannot enter the system by connecting to that particular group, or access resources as a member of that group.

RACF allows you to specify a future date for a REVOKE to occur (at both the system and the group level). You can also specify a future date to remove the REVOKE attribute by using the RESUME operand on the ALTUSER command.

Only the owner of a user’s profile (or a user who has the SPECIAL attribute) can assign the REVOKE attribute.

The GRPACC (Group Access) Attribute

If a user has the GRPACC attribute, any group data set profiles that the user defines to RACF (through either the ADSP attribute, the PROTECT parameter on the DD statement, or the ADDSD command) are automatically made accessible to other users in the group if the user defining the profile is a member of that group. The group whose name is used as the high-level qualifier of the data set name is given UPDATE authority to the data set. Note that, if the defining user does not

have the GRPACC attribute, and profile modeling is not being used, the user must use the PERMIT command to allow the group to access the group data set.

A user to whom you assign the GRPACC attribute at the *user* level has this attribute in all of the groups of which the user is a member. If a user has the GRPACC attribute at the *group* level, the attribute applies to only the group in which the user has the attribute.

You should assign the GRPACC attribute with care, especially if the RACF user to whom you are assigning the attribute is allowed to RACF-protect group data sets in several groups. This user could unintentionally authorize groups to access a group data set to which they should not have access.

Only the owner of a user's profile (or a user who has the SPECIAL attribute) can assign the GRPACC attribute.

Notes:

1. The use of automatic modeling (for example, the MODEL operand in user and group profiles) provides more flexibility than the GRPACC attribute.
2. You can provide more flexible coverage for all users, in some resource classes, by using appropriate &RACGPID entries in the global access checking table. For more information, see Table 18 on page 211.

The ADSP (Automatic Data Set Protection) Attribute

When a user has the ADSP attribute, RACF always automatically creates a discrete profile every time the user defines a permanent DASD or tape data set. (For tape data sets, the TAPEDSN and TAPEVOL options must be active.)

You can assign ADSP at the group level using the CONNECT command. If assigned at the group level, ADSP is in effect only when that group is the user's current connect group.

If generic profile checking is active, you should consider removing the user's ADSP attribute. You can do this on a user-by-user basis with the ALTUSER command, or for an entire installation by using the NOADSP operand on the SETROPTS command.

A data set created under ADSP is accessible only to the user who created it, unless other users or groups are added to the access list (such as through the PERMIT command, the GRPACC user attribute, or modeling), or if global access checking allows the access.

Only the owner of a user's profile (or a user who has the SPECIAL attribute) has control over the ADSP attribute.

Attention

A DASD data set is defined to RACF at allocation. If the data set disposition is changed at deallocation (through dynamic deallocation), the change is *not* reflected in the RACF database. For example, if the data set disposition is DELETE at allocation and KEEP at deallocation, the data set is not automatically RACF-protected. However, RACF performs generic profile checking if you have activated this option for the DATASET class by specifying GENERIC(DATASET) on the SETROPTS command.

The RESTRICTED Attribute

You can prevent RACF users from gaining access to protected resources they are not specifically authorized to access by assigning the RESTRICTED attribute on the ADDUSER or ALTUSER command. See“Defining Restricted User IDs” on page 87 for more information.

Only the owner of a user’s profile (or a user who has the SPECIAL attribute) can assign the RESTRICTED attribute.

User Attributes at the Group Level

You can specify the SPECIAL, AUDITOR, and OPERATIONS user attributes at the group level by using the CONNECT command. When you specify these attributes at the group level, they are identified as group-SPECIAL, group-AUDITOR, and group-OPERATIONS to distinguish them from attributes at the system level.

Group attributes are indicated in the description of the user-to-group connection in the user profile. Unless list-of-group checking is active, group attributes are in effect for the user only when the user is connected to the group during a batch job or terminal session.

If list-of-groups checking is active, then, regardless of which group the user is logged on to (the current connect group), RACF recognizes the user’s group-related attributes in the user’s other connect groups. (It is as though the user was logged on to each group at the same time.) For more information on list-of-groups checking, see “Activating List-of-Groups Checking (GRPLIST Option)” on page 116.

When you initially define a new user, the user’s connection to the default group does not indicate any group-related attributes. You can then use the CONNECT command to define the user’s group attributes within the default group.

The Scope of Authority for the Users with Group-Level Attributes

The authority of the group-SPECIAL, group-AUDITOR, and group-OPERATIONS users is limited to the resources that are within the scope of the group. Resources that are within the scope of the group include the following (see Figure 5 on page 82 and Figure 6 on page 83):

- Resources owned by the group (for example, GROUP1.DATA owned by GROUP1)
- Resources that are owned by users who are owned by the group (for example, USER2.DATA owned by USER2 who is owned by GROUP1)
- Resources that are owned by subgroups that are owned by the group (for example, GROUP2.DATA owned by GROUP2, which is owned by GROUP1)
- Resources that are owned by subgroups that are owned by subgroups, owned by the group, and so on (for example, GROUPZ.DATA owned by GROUPZ, which is owned by GROUP2, which in turn is owned by GROUP1)

Note that the scope of the group does *not* extend to the following resources:

- Resources that are owned by groups that are owned by users who are owned by the group (for example, GROUPY.DATA owned by GROUPY which is owned by USER2 who is owned by GROUP1)
- Resources owned by users who are, in turn, owned by users who are owned by the group (for example, USER6.DATA owned by USER6 who is, in turn, owned by USER5 who is owned by GROUP2)

Groups and Users

By establishing the group structure so that subgroups are owned by their superior groups, the authority of the group-SPECIAL, group-OPERATIONS, and group-AUDITOR user can be made to percolate down through the group tree structure as far as the security administrator desires. When a user's attribute percolates down from a group to which the user is connected with the group attribute, the user's authority in the subgroups is the same as if the user was connected directly to the subgroups with the group attribute.

Note: The data security monitor (DSMON) produces a group tree report that lists, for each requested group, all of its subgroups, all of the subgroups' subgroups, and so on. This report can be very useful in checking to which subgroups the authority of the group-SPECIAL, group-OPERATIONS, or group-AUDITOR applies. For more information on the group tree report, see *z/OS SecureWay Security Server RACF Auditor's Guide*.

The limits of the security administrator, group administrator, auditor, and operations personnel authority at the group level are described in Table 7. (Of course, these users continue to have whatever authorities they possess from other sources, such as ownership, that are not covered by their group-level authorities.)

Table 7. Scope of Authority for User Attributes at the Group Level

Resource	Attribute, User, and Authority
Data Sets	**Group-SPECIAL Attribute:** A user with the group-SPECIAL attribute has full authority to work with: • Data set profiles that are owned by the group • Data set profiles that have a high-level qualifier that is the same as the group identifier • Data set profiles that are owned by users or groups that are owned by the group • Data set profiles that have a high-level qualifier that is a user or group identifier owned by the group The group-SPECIAL user can also define data set profiles with a high-level qualifier that is the group identifier or a user or group identifier owned by the group. **Group-AUDITOR and Group-OPERATIONS Attributes:** A user with the group-AUDITOR or group-OPERATIONS attribute can perform all of the functions of an auditor or operator, but is limited to the same subset of data sets as the user with the group-SPECIAL attribute.
General Resources	**Group-SPECIAL Attribute:** A user who has the group-SPECIAL attribute has full authority to work with: • Resource profiles that are owned by that group • Resource profiles belonging to users or groups that are owned by the group To create new resources, the user must have the CLAUTH attribute in the applicable class. **Group-AUDITOR and Group-OPERATIONS attributes:** A user who has the AUDITOR or OPERATIONS attribute can perform all of the functions of an auditor or operator, but is limited to the same above subset of resources as the user with the group-SPECIAL attribute.

Table 7. Scope of Authority for User Attributes at the Group Level (continued)

Resource	Attribute, User, and Authority
Users	**Group-SPECIAL Attribute:** A user with the group-SPECIAL attribute has full authority to work with: • User profiles that are owned by the group • User profiles that are owned by a subgroup that is owned by the group, by a subgroup that is owned by a subgroup that is owned by the group, and so on The group-SPECIAL user must have the CLAUTH attribute in a class in order to give the CLAUTH attribute to another user in that class. The group-SPECIAL user cannot give a user the SPECIAL, AUDITOR, or OPERATIONS attribute at a system level, but can assign these attributes at the group level. To create new users, the group-SPECIAL user must have the CLAUTH attribute in the USER class. **Group-AUDITOR Attribute:** A user who has the group-AUDITOR attribute can perform all of the functions of an auditor, but is limited to the same subset of users as the user with the group-SPECIAL attribute.
Groups	**Group-SPECIAL Attribute:** A user who has the group-SPECIAL attribute has authority over that group, over subgroups owned by that group, and so on. The group-SPECIAL user can connect any user to, or remove any user from, any group that is included in this authority.

The following two figures show the scope of authority of a group-SPECIAL user. Figure 5 on page 82 shows a typical authority structure containing three major groups: Groups 1, 2, and 3.

Figure 6 on page 83 shows the addition of a new element: a new user, USER1, is connected to Group 1. The resultant authority USER1 receives as a group-SPECIAL user is highlighted (the non-shaded area) in part 2 of this figure.

USER1 has authority to the profiles in the non-shaded area for the reasons summarized in Table 7 on page 80. USER1 does *not* have authority to any of the resources in the shaded area for the following reasons:

- GROUP1 does not own IBMUSER, GROUP3, USER3, or USER4.
- GROUP1 does not own GROUPY.
- Neither GROUP1 nor GROUP2 own USER6.
- USER3.DATA is not owned by a user who is owned by GROUP1.
- USER4.DATA is not owned by a user who is owned by GROUP1. USER1 cannot display the profile information for this data set with LISTDSD, even if USER2, for example, is in its access list. (However, if USER1 runs the IRRUT100 utility, RACF informs USER1 that USER2 is in the access list of USER4.DATA.)
- U4A is not a general resource that is owned by a user who is owned by GROUP1.

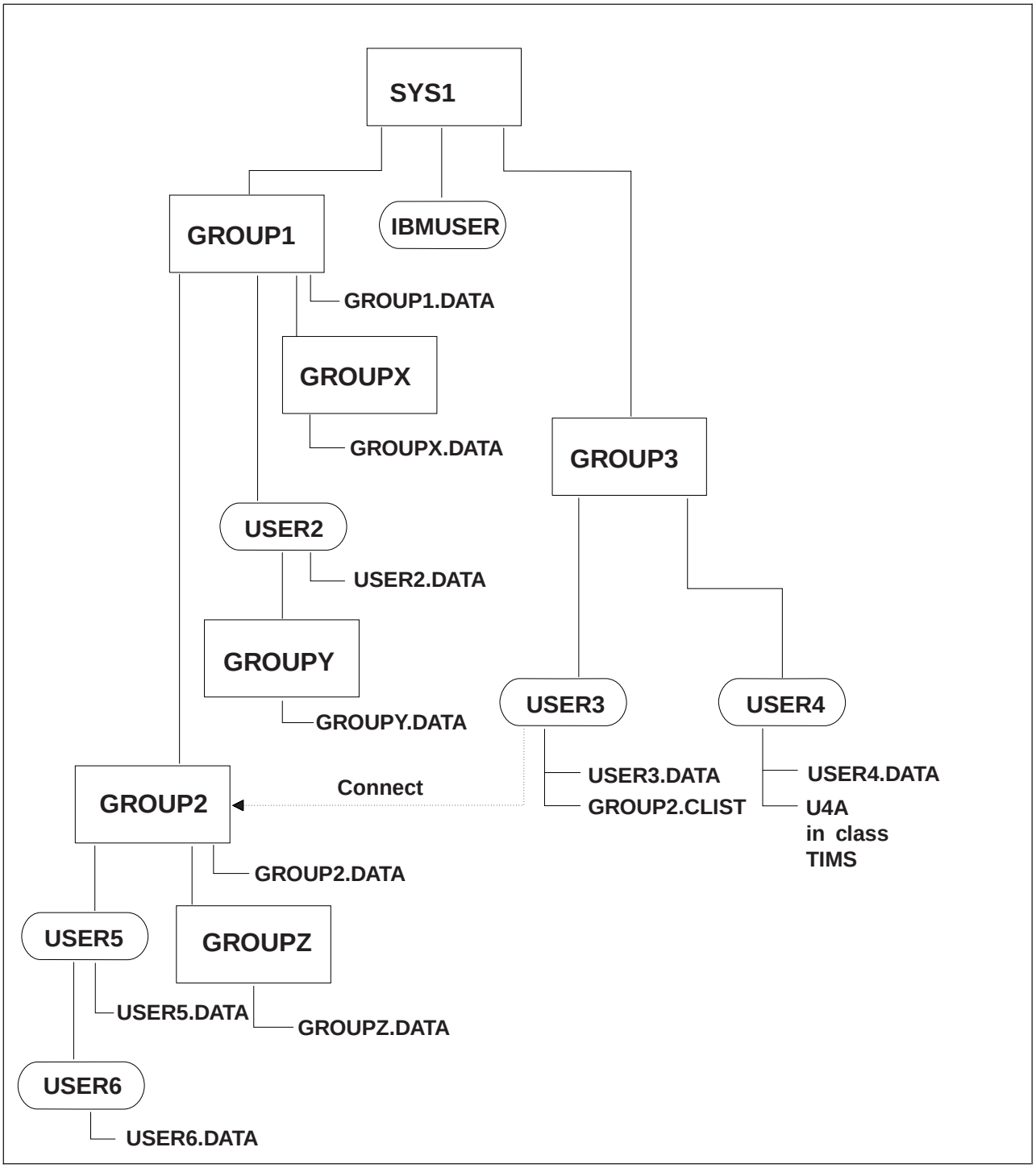


Figure 5. Group-Level Authority Structure

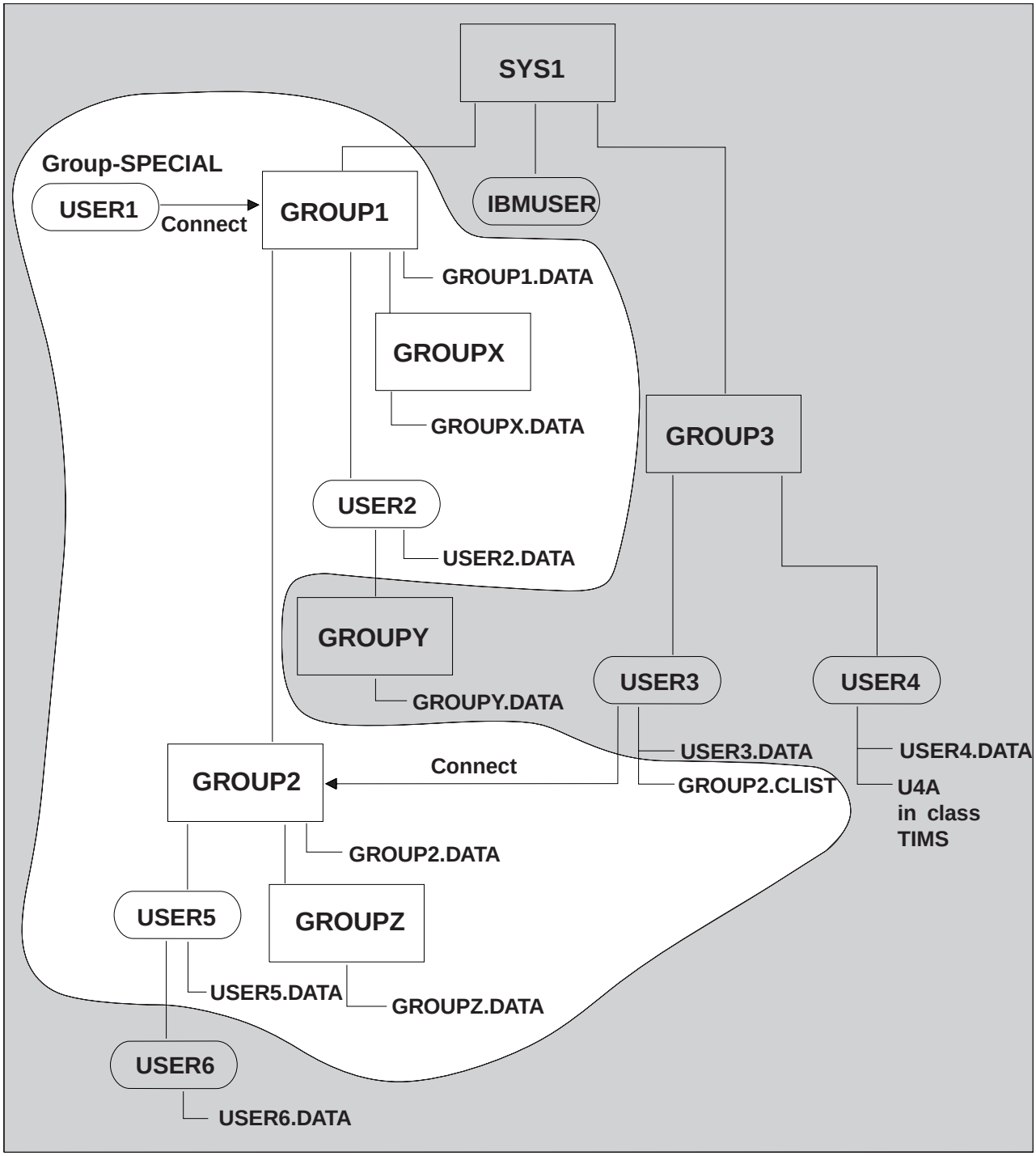


Figure 6. Scope of Authority of a Group-SPECIAL User

Suggestions for Assigning User Attributes

When defining users to RACF with the ADDUSER command, or when modifying user attributes with the ALTUSER command, RACF security and group administrators should assign:

- SPECIAL, AUDITOR, and OPERATIONS attributes to only those users responsible for administering RACF on a system-wide basis
- CLAUTH attributes to only those users who are responsible for defining other users and general resources
- RESTRICTED attribute to only those user IDs that should not gain access to protected resources they are not specifically authorized to access

Note: You cannot assign the ADSP attribute to a user who allocates space for data sets that do not meet the RACF or installation naming conventions.

Verifying User Attributes

The data security monitor (DSMON) generates reports that describe the current status of the data security environment at your installation. Two of these reports, the selected user attribute report and the selected user attribute summary report, are useful for verifying the attributes that you have assigned.

The selected user attribute report lists all RACF users with the SPECIAL, OPERATIONS, AUDITOR, or REVOKE attributes and specifies whether they possess these attributes on a system-wide (user) or group level. You can use this report to verify that only those users whom you want authorized to perform certain functions have been assigned the corresponding attribute.

The selected user attribute summary report shows the number of installation-defined users and totals for users with the SPECIAL, OPERATIONS, AUDITOR, and REVOKE attributes, at both the system and group level. You can use this report to verify that the number of users with each of these attributes, on either a system or group level, is the number that your installation wants.

Default Universal Access Authority (UACC)

Each user who is connected to a group is assigned a default universal access authority (UACC) of NONE, READ, UPDATE, CONTROL, or ALTER. You can specify this default UACC on the ADDUSER, ALTUSER, or CONNECT command. If you do not specify a value for UACC, RACF uses NONE as a user's default universal access authority.

RACF uses this default UACC for all new resources that a user defines while connected to the specified default group as follows:

- When a user issues the ADDSD command to define a new data set profile and does not specify a value for the UACC operand, RACF uses the default UACC as the UACC for the profile unless profile modelling is used.
- When a user issues the RDEFINE command to define a new general resource profile and does not specify a value for the UACC operand, RACF uses the default UACC as the UACC for the profile unless a value for UACC is specified in the class descriptor table (CDT).

For more information on using the UACC operand on the ADDUSER, ALTUSER, and CONNECT commands, see *z/OS SecureWay Security Server RACF Command Language Reference*.

Assigning Security Categories, Levels, and Labels to Users

You can assign security categories and security levels to users and sensitive resources. You can also assign security labels, which are a combination of security levels and security categories, and are more easily maintained, to users and

sensitive resources. User profiles and resource profiles that have been assigned a security label need not be changed if the definition of a security label is changed.

A security category is an installation-defined name corresponding to department or area within an organization that has similar security requirements. A security level is an installation-defined name that is associated with a number in the range 1 through 254.

If security levels and categories are being used (the SECDATA class is active), and security labels are *not* being used (the SECLABEL class is not active), RACF takes the following steps when a user requests access to a resource that has a security category or a security level associated with it:

1. If the resource has a SECLEVEL, RACF compares the security level of the user with the security level of the resource. If the resource has a higher security level than the user, RACF denies the request.

   For a terminal session, the security level that RACF uses for the user is the lower of the user's SECLEVEL and the terminal's SECLEVEL. Thus, if the terminal has a SECLEVEL of 50 and the user has a SECLEVEL of 100, the user cannot access, through that terminal, any data that has a SECLEVEL of over 50. RACF then proceeds to the category check.

   If the resource does not have a SECLEVEL, RACF proceeds to the category check in Step 2.
2. RACF compares the list of security categories in the user's profile with the security categories in the resource profile. If the user's security level is high enough to access the resource, RACF compares the list of security categories in the user's profile with the list of security categories in the resource's profile. If RACF finds any security category in the resource profile that is not in the user's profile, RACF denies the request. If RACF does not deny the request, RACF continues with authorization processing. If there are no categories in the resource profile, RACF continues with authorization processing.

If your installation has activated the SECLABEL class, and a user requests access to a resource that has a security label associated with it, RACF *ignores* any security level or security categories that are specified in the resource profile. Instead, RACF performs security label authorization checking, which involves the security levels and categories that are used to define the security labels of the resource and the user.

You can use security labels as a simple replacement for security levels and categories, with the same access authority requirements, or you can use SETROPTS options such as MLACTIVE and MLS to set up a more rigorous security environment. For more information on how the SETROPTS options change the effects of security labels, see "Security Label Authorization Checking" on page 647. For more information on security classification of users and data, see "Chapter 4. Classifying Users and Data" on page 95.

Limiting When a User Can Access the System

Installations can limit a user's ability to log on by limiting:

- The user's ability to log on to the system to certain days of the week, and certain hours within each day
- The use of individual terminals (in the TERMINAL class only) to certain days of the week, and certain hours within each day

To limit the times during which a user can enter the system, use the WHEN operand on the ADDUSER and ALTUSER commands. For example, to specify that USER12 can enter the system only on weekdays between the hours of 7:00 a.m. and 5:00 p.m., enter:

```
ADDUSER USER12 WHEN(DAYS(WEEKDAYS) TIME(0700:1700))
```

Similarly, to control when users can access the system from a specific terminal, specify the WHEN operand on the RDEFINE and RALTER commands for the appropriate profile. For example, to specify that terminal TRM07C can be used at any time during the week, but not at all during the weekend, enter:

```
RDEFINE TERMINAL TRM07C WHEN(DAYS(WEEKDAYS))
```

Note that on the RDEFINE command, TIME(ANYTIME) is the default.

The WHEN operand on these commands (for both users and terminals) allows you to specify individual days and specific times within these days.

RACF also provides support for installations that have terminals in different time zones by associating with each terminal its location relative to the local time where the processor complex on which RACF is executing is located.

Note: RACF does not provide any specific support for daylight savings time. If the installation changes the value of the local time (as given by the TIME macro) to accommodate daylight savings time, RACF automatically adjusts its time calculations accordingly. However, if any terminals are located in an area that does not follow the same time adjustment, you must adjust the terminal information.

For more information on the WHEN operand, see the command descriptions in *z/OS SecureWay Security Server RACF Command Language Reference*.

Time-of-Day and Day-of-Week Checking for Users and Terminals

The time and day-of-week checking for users and terminals applies when users log on to terminals from products such as TSO/E, IMS, CICS, and NetView (beginning with Release 2), but does not apply to batch jobs or started tasks.

User verification processing includes the following time/day-of-week checks:

1. The user's authority to use the specific terminal. If the profile protecting the terminal does not have any time or day-of-week information, the user can log on. If there is time and day-of-week information, RACF calculates the time-of-day at the location of the terminal from which the user is logging on (if that time is different from the time-of-day at the location of the processor complex), and checks whether the terminal can be used at this time on this day of the week.
2. The user's authority to access the system. If the user's profile does not have any time or day-of-week information, the user can log on. If there is time and day-of-week information, RACF calculates the time-of-day at the location of the terminal from which the user is logging on (if that time is different from the time-of-day at the location of the processor complex), and checks whether the user can enter the system.

Defining Protected User IDs

You can define a protected user ID by assigning both the NOPASSWORD and NOOIDCARD attributes through the ADDUSER or ALTUSER command. Protected user IDs are protected from being used to logon to the system, and from being

revoked through incorrect password attempts. However, they can be revoked using the ALTUSER (*userid*) REVOKE command, and through inactivity. If revoked, protected user IDs can be activated using the ALTUSER (*userid*) RESUME command.

A protected user ID cannot be used to enter the system by any method that uses a supplied password, such as TSO logon, CICS signon, z/OS UNIX `rlogin`, or batch job submissions when a password is specified using the PASSWORD parameter of the JOB statement. Before assigning the PROTECTED attribute to a user ID, you should ensure that the user ID will not be used in any situation where specification of a password is required.

You may wish to assign protected user IDs to z/OS UNIX, and to the UNIX daemons, started procedures, applications, servers or subsystems associated with z/OS UNIX, to minimize their exposure to inadvertent or malicious misuse or revocation. Surrogate-submitted batch jobs can use protected user IDs. See “Using Protected User IDs for Batch Jobs” on page 444 for more information. Protected users can be associated with started procedures defined in the STARTED class (preferred method) or in the started procedures table (ICHRIN03). For more information, see “Assigning RACF User IDs to Started Procedures” on page 143.

The following example shows the ALTUSER command used to assign the PROTECTED attribute to an existing user ID.

```
ALTUSER SERVER8 NOPASSWORD
```

A protected user ID will have the PROTECTED attribute displayed in the output of the LISTUSER command.

Attention

If you share the RACF database, protected user IDs *may* be used to attempt logon from systems running OS/390 releases prior to OS/390 Version 2 Release 8. This may result in protected user IDs being revoked through malicious or inadvertent incorrect password attempts from downlevel systems. To avoid this, you should make sure that z/OS, or OS/390 Version 2 Release 8 or later, is installed on all shared systems before defining protected user IDs.

Do not issue ADDUSER and ALTUSER commands specifying the PASSWORD/NOPASSWORD or OIDCARD/NOOIDCARD operands to administer protected user IDs from downlevel systems. If an existing protected user ID is administered from a downlevel system using the ALTUSER command with these operands, the user ID may lose its PROTECTED attribute.

A LISTUSER command issued for a protected user ID from a downlevel system will not display the PROTECTED attribute.

Defining Restricted User IDs

You can define a restricted user ID by assigning the RESTRICTED attribute through the ADDUSER or ALTUSER command. Restricted user IDs cannot be used to access protected resources they are not specifically authorized to access. Access authorization for restricted user IDs bypasses global access checking. In addition, the UACC of a resource and an `ID(*)` entry on the access list are not used to enable a restricted user ID to gain access.

The RESTRICTED attribute can be added to shared user IDs, such as PUBLIC and ANONYMOS, that are assigned by application servers that allow users to enter the system without identifying themselves. Without the RESTRICTED attribute, users that are assigned shared user IDs can gain access to any resource that has an `ID(*)` entry in the access list, UACC, or global entry that allows access.

The following example shows the ALTUSER command used to assign the RESTRICTED attribute to an existing shared user ID.

```
ALTUSER ANONYMOS RESTRICTED
```

A restricted user ID has the RESTRICTED attribute displayed in the output of the LISTUSER command.

The RESTRICTED attribute has no effect on access authorization checking for z/OS UNIX resources, such as hierarchical file system (HFS) files. These resources do not have RACF access lists; they use permission bits to authorize only three types of users: the resource owner, group members, and all others. Therefore, permission bits are used to authorize access to these resources for all users, including restricted users.

Using Restricted User IDs for Digital Certificate Users

Users who identify themselves by supplying a digital certificate that is not registered to RACF are eligible for certificate name filtering. These users may be assigned a RACF user ID on your system if there is an applicable name filter in effect. See “Certificate Name Filtering” on page 531 for more information.

To prevent users who gain access through certificate name filtering from accessing protected resources they are not specifically authorized to access, you should assign restricted user IDs to each user ID associated with a certificate name filter.

Attention

If you share the RACF database, restricted user IDs *may* be used to access resources they are not specifically authorized to access on downlevel systems that do not support restricted user IDs. To avoid this, you should make sure that certificate name filtering support is installed on all shared systems before defining restricted user IDs.

A LISTUSER command issued for a restricted user ID from a downlevel system will not display the RESTRICTED attribute.

Summary of Steps for Defining Users

This summary presents the steps required by RACF and related IBM licensed programs to define users to RACF. Your installation may require additional steps, depending on your security policy and the products you have installed.

1. Prepare to create the user profile as follows:
   - Decide which default connect group to assign to the user. If a group profile does not yet exist for the group, create the group using the procedure described in “Summary of Steps for Defining a RACF Group” on page 59.
   - Decide which user ID to assign to the user.
   - Decide which user or group is to be the owner of the user profile. (If the owner is a user, give him or her the information needed to manage the new profile.)

- Decide what initial password to assign to the user. (If you do not specify a password, the new user's default group name becomes the new user's initial password. You might prefer to specify a non-trivial password.)
- Determine if the user's access to the system should be limited to certain days of the week, hours of the day, or both.
- Decide which user attributes (such as SPECIAL or AUDITOR) the user should have, and whether the user attributes should be limited to the scope of a group (group-SPECIAL or group-AUDITOR).
- If security labels are used, decide which security label to assign to the user.
- Decide whether the user can establish user ID associations to enable password synchronization and command direction between user IDs. See "Chapter 10. The RACF Remote Sharing Facility (RRSF)" on page 355 for more information.
- If DFSMS is installed, work with the storage administrator to do the following:
  - Determine the initial values in the user's DFP segment.
  - Determine which DFP resources the user should have access to.
- Determine which primary and secondary languages the user should have (if they should be different from the installation defaults set by the SETROPTS command).

2. If you want to authorize the user to establish an extended MCS console session, work with the system operations planner to determine the initial values in the user's OPERPARM segment. For more information, see "The OPERPARM Segment in User Profiles" on page 68 and *z/OS MVS Planning: Operations*.
3. If the user is a CICS user, work with the CICS administrator to do the following:
   - Determine the initial values in the user's CICS segment.
   - Determine which primary and secondary languages the user should have (if they should be different from the CICS-specified installation defaults).

     Note: CICS does not check the installation defaults set by the SETROPTS command.
   - Determine the CICS resources to which the user should have access.
   - For other more specific information, see *CICS RACF Security Guide*.
4. Work with the APPC administrator to do the following:
   - Determine the initial values in the user's WORKATTR segment.
   - Determine which APPC/MVS resources the user should have access to.
5. Create the user profile.

   Note: This can be done using any of the following methods:
   - Issuing the ADDUSER command.
   - Enrolling the user through the TSO/E Information Center Facility (ICF) panels.

     For more information about administering the Information Center Facility, see *z/OS TSO/E Administration*.

   Here is an example of using the ADDUSER command to create a user profile. Suppose you want to create a user profile for user Steve H., a member of Department A. You want to assign the following values:
   - STEVEH for the user ID
   - DEPTA for the default connect group
   - DEPTA for the owner of the STEVEH user profile
   - R3I5VQX for the initial password

- Steve H. for the user's name

Steve H. does not require any of the user profile segments except TSO. The TSO segment values that you want to set to start with are 123456 for the account number and PROC01 for the logon procedure.

To create a user profile with these values, enter:

```
ADDUSER STEVEH DFLTGRP(DEPTA) OWNER(DEPTA) NAME('Steve H.')
       PASSWORD(R315VQX) TSO(ACCTNUM(123456) PROC(PROC01))
```

6. Create a "top" generic profile for the user in the DATASET class using the ADDSD command.

   For example, if the user's user ID is STEVEH, enter:

   ```
   ADDSD 'STEVEH.**' UACC(NONE)
   ```

7. If users at your installation manage their own resource profiles, give them the information they need. For example, they might need to use portions of *z/OS SecureWay Security Server RACF General User's Guide*.
8. If the user is to define general resource profiles, (as, for example, an administrator might), give the user the CLAUTH attribute in the appropriate classes and the information needed for working with those profiles, for example, the JESSPOOL class.

   Note: If the SETROPTS GENERICOWNER option is in effect, you must create a "top" profile for the user in the JESSPOOL class, make the user the owner of the profile, and give the user CLAUTH(JESSPOOL). For more information, see "Letting Users Create Their Own JESSPOOL Profiles" on page 473 and "Defining Profiles for SYSIN and SYSOUT Data Sets" on page 471.
9. If needed, give the user access to RACF-protected resources. This can be done using one or both of the following:
   - Connect the user to groups that have the same access requirements as this user, using the CONNECT command.

     For example, to allow user STEVEH to have access to his department's resources (that is, to resources belonging to group DEPTA), enter:

     ```
     CONNECT STEVEH GROUP(DEPTA) OWNER(DEPTA)
     ```

     By default, the command gives USE authority to STEVEH.
   - If the user requires specific access to RACF-protected resources (beyond that permitted by connecting the user to groups), give the user the access required, using the PERMIT command.

     Consider the following:

     – If the user is a TSO user, remember the necessary TSO resources (such as TSOPROC).

     – If data sets are managed by SMS, remember the MGMTCLAS and STORCLAS classes.

     For example, to give user STEVEH permission to use a customized TSO logon procedure called CUSTPROC (whose profile in the TSOPROC general resource class has already been defined with a universal access of NONE), enter:

     ```
     PERMIT CUSTPROC CLASS(TSOPROC) ID(STEVEH) ACCESS(READ)
     ```

Summary of Steps for Deleting Users

This summary presents the steps required by RACF and related IBM licensed programs to delete users from RACF. Your installation may require additional steps, depending on your security policy and the products you have installed.

1. To prevent the user from entering the system, revoke the user ID:

   ```
   ALTUSER userid REVOKE
   ```

2. If the user is already logged onto the system, or has a job running on the system, ask the system operator to examine any logons (or jobs) for the user and cancel those that should not be allowed to continue.
3. Use the RACLINK LIST command to see if the user has any user ID associations defined. If so, use the RACLINK UNDEFINE command to delete them. You can't delete a user ID that has any associations defined. See "Chapter 10. The RACF Remote Sharing Facility (RRSF)" on page 355 for more information.
4. Find all of the data sets associated with this user (that is, data sets for which the user's user ID is the high-level qualifier of the data set name) and take the following steps:
   a. Delete or rename (with a new high-level qualifier) the user's user data sets. If you rename or delete a data set that is protected by a discrete profile, the discrete profile is also renamed or deleted.

      Note: You can do this using the DATA SET LIST utility of ISPF.

   b. Identify all of the remaining (generic) data set profiles, create new ones modeled on them if needed, and then delete the remaining profiles.

      Attention
      Make sure that you do not delete an old profile unless it is no longer needed.

      Notes:

      1) You can use the following SEARCH command to identify the user's data set profiles:

         ```
         SEARCH MASK(userid.) CLIST('LISTDSD ' ' ALL')
         ```

         As specified, the CLIST operand generates a CLIST that you can run to list all of the information in the data set profiles. This can help you assess whether to use the profiles as models.

      2) You can use the FROM operand on the ADDSD command to create new profiles modeled on the old profiles.

      If the user has profiles in other classes (such as the JESSPOOL, JESJOBS, and NODES classes) that might have the user's user ID in their profile names, use the FILTER operand on the SEARCH command. For example:

      ```
      SEARCH CLASS(classname) FILTER(**.userid.**)
             CLIST('RDELETE classname')
      ```

5. To research the following steps, use the IRRRID00 utility to list the occurrences of the user ID in the RACF database. For information, see "Using the RACF Remove ID Utility (IRRRID00)" on page 342.

6. If the user is the owner of group data set profiles (the user's user ID was specified on the OWNER operand on the ADDSD or ALTDSD command for the group data set profile), decide which user or group is to be the new owner of the group data set profiles.

   Note: If the user is the owner of any group data set profiles, specify the new owner on the OWNER operand of the REMOVE command.
7. If the user is a TSO user and has a SYS1.UADS entry, work with the TSO administrator to delete the entry.
8. If the user is a CICS user and has an entry in the CICS signon table, work with the CICS administrator to delete the entry.
9. Remove the user from any access lists in which the user's user ID is specified.

   Note: To do this, use the DELETE operand on the PERMIT command.

   For example, suppose user ELVIS has update permission to a set of data sets defined in the PROJA.** profile. To remove ELVIS from the profile's access list, enter:

   ```
   PERMIT 'PROJA.**' ID(ELVIS) DELETE
   ```
10. If the user owns any RACF profiles, change the OWNER field of the profile.

    Note: To do this, use the appropriate command for changing profiles, such as ALTUSER or RALTER.
11. After all occurrences of the user ID are deleted from the RACF database, use the DELUSER command to delete the user profile.

    For example, to delete the profile for user ELVIS, enter: `DELUSER ELVIS`

General Considerations for User ID Delegation

This section discusses things to consider for delegating administrative tasks to other users.

- In general, centralize first, delegate later.
- Consider the trade-offs:
  - Should one user handle all of the administration workload?
  - Should many users all be learning RACF simultaneously?
- RACF groups (not users) should own resource profiles.
- Authorize groups rather than users to resource profiles.
- Delegate power (group-SPECIAL) with care.
- Have "standby" SPECIAL and OPERATIONS user IDs for emergency situations.

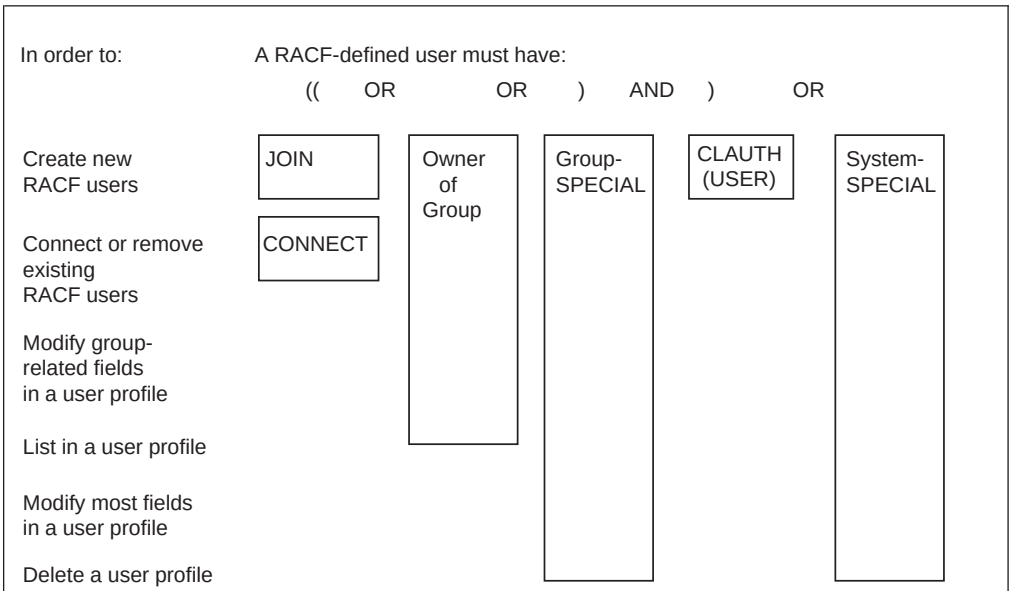
  Note: The password for the "standby" user IDs should be kept under lock and key.
- After control has been given, it is difficult to take it away again.
- Group-SPECIAL is the most powerful authority a user can have at the group level.
  - Group-SPECIAL enables the user to use more commands.
  - Group-SPECIAL also percolates to other groups, as far as the scope of the group allows.

Choose the best *current* option for your installation.

- For authority over a *single* group of resources based on protection objectives, use JOIN and CLAUTH(USER).
- For authority over one or more groups of resources based on protection objectives and scope of the group, use group-SPECIAL and CLAUTH(USER).

Note: The group-SPECIAL attribute allows password resetting for user IDs within the group whereas JOIN does not.

Figure 7 shows delegating authority in another way.



A combination of authorities is required to create new RACF users.

For example, to create a new RACF user, the creating user must have at least one of the following *and* have the CLAUTH(USER) attribute:

- JOIN group authority in the new user's default group

  or
- Be the owner of the new user's default group

  or
- Have group-SPECIAL in the new user's default group

  or
- Have SPECIAL

Figure 7. Delegating Authority (User Profiles)

Chapter 4. Classifying Users and Data

This chapter contains in-depth information on using security levels, categories, and labels to classify users and data.

Security Classification of Users and Data

Security classification of users and data allows installations to impose additional access controls on sensitive resources. Each user and each resource can have a security classification in its profile. You can choose among the following:

- Security levels, security categories, or both
- You can use security labels, which are a combination of security levels and security categories, and are easier to maintain

A *security level (SECLEVEL)* is an installation-defined name that corresponds to a numerical security level (the higher the number, the higher the security level).

A *security category (CATEGORY)* is an installation-defined name that corresponds to a department or an area within an organization in which the users have similar security requirements.

Security Classification

A *security label (SECLABEL)* is an installation-defined name that corresponds to a security level and zero or more security categories.

This section discusses security levels and security categories first. Security labels are discussed later (see “Understanding Security Labels” on page 101).

Effect on RACF Authorization Checking

Security classification processing takes place after global access checking (if active), but before RACF checks the standard access list. If global access checking does not allow access to the resource, RACF does security classification processing for any resource that is protected by a profile that has security category or security level data. (For information on global access checking, see “Setting Up the Global Access Checking Table” on page 206. For a complete list of the sequence of checks that RACF makes to grant or deny access to a resource, see “Authorization Checking for RACF-Protected Resources” on page 633.)

Attention

Because RACF performs global access checking before many of the other kinds of access authority checks, such as security label checking or access list checking, global access checking might allow access to a resource you are otherwise protecting. To avoid a security exposure to a sensitive resource, do not create an entry in the global access checking table for a resource protected by a profile that contains a security level, security category, or security label (if the security label in the profile is SYSLOW, a global access checking table entry with an access authority of READ can be created). See “Authorization Checking for RACF-Protected Resources” on page 633.

Security Levels and Security Categories

Security classification processing consists of a two-step checking process that occurs when RACF is processing an authorization request. (Note that the SECDATA class must be active, the SECLABEL class must not be active, and the protecting resource profile must have security levels or security categories.)

1. RACF compares the security level of the user with the security level of the resource. If the resource has a higher security level than the user, RACF denies the request.

   For a terminal session, the security level that RACF uses for the user is the lower of the user’s SECLEVEL and the terminal’s SECLEVEL. Thus if the terminal has a SECLEVEL of 50 and the user has a SECLEVEL of 100, the user cannot access, through that terminal, any data that has a SECLEVEL of over 50.
2. RACF compares the list of security categories in the user’s profile with the list of security categories in the resource’s profile. If RACF finds any security category in the resource profile that is not in the user’s profile, RACF denies the request. If RACF does not deny the request, RACF continues with authorization processing. If there are no categories in the resource profile, RACF continues with authorization processing.

Security Labels

Security label authorization checking is dependent on the concept of controlling user access to resources on the basis of three factors:

1. The sensitivity of the data that the resource contains
2. The user’s authorization to access information at that level of sensitivity
3. The purpose for which the user is attempting to access the resource

The security administrator indicates the sensitivity of the data in the resource as well as the authorization of the user by assigning appropriate security labels in the resource or user profile.

Security label authorization checking involves comparing the user's security label with the security label of the resource. A user who lacks sufficient authorization is prevented from accessing information in the resource.

Three types of authorization checks are used to determine security label authorization:

- **Read Only (R/O):** A user is attempting to read information from a resource
- **Write Only (W/O):** A user is attempting to write information to a resource (with no reading)
- **Read and Write (R/W):** A user is attempting to access a resource for the purpose of both reading and writing

For more information, see "Security Label Authorization Checking" on page 647.

Understanding Security Levels and Security Categories

When RACF is first installed, security classification of users and data is inactive. To use security levels and categories, activate the SECDATA class (but not the SECLABEL class).

You can choose to use one or both parts of security classification processing. To use security level checking, you must define a profile in the SECDATA general resource class with the name SECLEVEL. To use security category checking, you must define a profile in the SECDATA general resource class with the name CATEGORY. The installation names for security categories and security levels are then defined as members of these profiles (in a manner similar to the global access table entries). You maintain the member entries by using the ADDMEM operand on the RDEFINE command and the ADDMEM and DELMEM operands on the RALTER command.

In the CATEGORY profile, the member entries are the names of the security categories. In the SECLEVEL profile, each member entry consists of a security level name followed by its associated security level number.

Note: You cannot define a SECLEVEL for a SECLEVEL profile in the SECDATA class. As a result, RACF does not perform security level checking when determining a user's authority to access a SECLEVEL profile. Also, if you issue the RLIST SECDATA SECLEVEL command to display a SECLEVEL profile, RACF does not display values in the SECLEVEL or CATEGORY fields of the profile.

Defining and Maintaining Security Levels and Security Categories

To use security levels and categories, take the following steps:

1. Define the SECLEVEL profile to the SECDATA class using the RDEFINE command.

   ```
   RDEFINE SECDATA SECLEVEL UACC(NONE)
   ```

2. Define security levels as members of the SECLEVEL profile in the SECDATA class.

   ```
   RALTER SECDATA SECLEVEL ADDMEM(seclevel-name/seclevel-number ...)
   ```

3. Define the CATEGORY profile to the SECDATA class using the RDEFINE command.

   ```
   RDEFINE SECDATA CATEGORY UACC(NONE)
   ```

4. Define categories as members of the CATEGORY profile in the SECDATA class.

   ```
   RALTER SECDATA CATEGORY ADDMEM(category-1 category-2 ...)
   ```

5. Assign security levels, security categories, or both, to users:

   ```
   ALTUSER userid SECLEVEL(security-level-name)
   ALTUSER userid ADDCATEGORY(category-name1 category-name2 ...)
   ```

6. Assign security levels, security categories, or both, to resources, for example:

   ```
   RALTER classname profile-name SECLEVEL(security-level-name)

   RALTER classname profile-name
          ADDCATEGORY(category-name1 category-name2 ...)
   ```

7. When you are ready to start using security levels and security categories, activate the SECDATA class:

   ```
   SETROPTS CLASSACT(SECDATA)
   ```

Example Showing an Error and Its Correction

The following example illustrates an error in setting up security levels and how it can be corrected:

1. Define a profile named CATEGORY with member entries:

   ```
   RDEFINE SECDATA CATEGORY UACC(READ) ADDMEM(ACCOUNTING)
   ```

   This command creates a profile named CATEGORY in the SECDATA class with the entry ACCOUNTING in its member list.

   The UACC specification means that anybody can list this profile, to determine what security category names the installation has defined.

2. Define a profile named SECLEVEL:

   ```
   RDEFINE SECDATA SECLEVEL UACC(READ)
   ```

   This command creates a profile named SECLEVEL in the SECDATA class.

3. Define members to SECLEVEL:

   ```
   RALTER SECDATA SECLEVEL ADDMEM(IMPORTANT/10,ROUTINE/75,CONFIDENTIAL/150)
   ```

   This command defines security level names and the associated security level numbers as members of the SECLEVEL profile. The members created are:

Security Level Name	Number
IMPORTANT	10
ROUTINE	75
CONFIDENTIAL	150

   However, you discover that you made an error when defining the members of the SECLEVEL profile. You really wanted IMPORTANT to have a higher security level value than ROUTINE. To change this value, see the following example.

4. Change a level number:

   ```
   RALTER SECDATA SECLEVEL ADDMEM(IMPORTANT/100)
   ```

   This command changes the security level number associated with IMPORTANT. The new member list is:

Security Level Name	Number
ROUTINE	75
IMPORTANT	100
CONFIDENTIAL	150

Attention

Any change to existing SECLEVEL or CATEGORY members may cause unexpected results, because the change is not reflected in existing user and resource profiles. Whenever you make such a change, RACF issues a warning message to remind you. Installations can use the SEARCH command to find profiles that require changes. However, because RACF keeps track of security levels by number, replacing an existing security level name does not affect the protection that the security level number provides. If you had defined the security levels shown in the preceding example and then replaced CONFIDENTIAL/150 with SECRET/150, a listing of a user or resource profile that included the security level 150 would show the new name. Because the security level number is the same, there is no need to change any resource or user profiles.

Note: The need to change many profiles when a security level or security category is changed can be avoided by using security labels instead.

CATEGORY and SECLEVEL Information in Profiles

The RACF commands for users, data sets, and general resources allow you to define and maintain security classification information. Some examples of commands with security category and security level information follow. (For complete information on these commands, see *z/OS SecureWay Security Server RACF Command Language Reference*.) The examples assume that the SECLEVEL and CATEGORY tables shown earlier have been defined.

Examples of Using Security Categories and Levels

The following examples show you how to use security categories and security levels to protect data sets.

1. Protect with security category and security level information.

   ```
   ADDSD 'WINTERS.PATTERNS' UACC(NONE) SECLEVEL(ROUTINE) ADDCATEGORY(ACCOUNTING)
   ```

   This example creates a discrete data set profile with a security level of ROUTINE and a security category of ACCOUNTING.
2. Modify the security level information in an existing data set profile.

   ```
   ALTDSD 'WINTERS.PATTERNS' SECLEVEL(CONFIDENTIAL)
   ```

   This command modifies the WINTERS.PATTERNS data set profile to have a security level of CONFIDENTIAL instead of ROUTINE.
3. Delete all security level and category information in an existing data set profile.

   ```
   ALTDSD 'WINTERS.PATTERNS' NOSECLEVEL DELCATEGORY(*)
   ```

   This command modifies the WINTERS.PATTERNS data set profile by deleting the security level information, as well as all category information.

Security Classification

In a user's profile, the security classification information is an access allowance, whereas in a resource profile, it is an access restriction. In this way, the security level test "passes" a user whose SECLEVEL is greater than or equal to that of the resource. A similar situation exists with security categories. The security category test "passes" a user if the user's profile contains every security category that is in the resource's profile. However, passing the security level and category tests does not allow the user to access the resource. The user must also pass any other existing test.

Security classification information in user and resource profiles can be updated at any time. However, changes made to a user's security classification while the user is logged on do not take effect during that session.

Note: Only users with the SPECIAL attribute can give another user, data set, or general resource a security level higher than they have, or a security category that they do not have themselves.

Converting from LEVEL to SECLEVEL

Many installations use the LEVEL field for their own implementation of security-level checking. To convert these profiles to use SECLEVEL instead, installations can use the SEARCH command to search for profiles that have a specified value in the LEVEL field. Installations can use the SEARCH command with the CLIST option to locate all data set profiles with certain LEVEL values, and convert them to use SECLEVEL instead.

Attention

Before converting from the use of LEVEL to SECLEVEL, all user profiles must have the appropriate SECLEVELs (if the SECDATA class is activated).

Deleting UNKNOWN Categories

If you delete a member from the CATEGORY profile, and that category is still specified in resource profiles, the resource profile listing (produced by the RLIST command, for example) shows an UNKNOWN category. To delete this category, enter the RALTER command with no category specified on the DELCATEGORY operand:

```
RALTER classname profile-name DELCATEGORY
```

To search for such profiles, enter the search command as follows:

```
SEARCH CLASS(classname) CATEGORY
```

Maintaining Categories in an RRSF Environment

RACF assigns an internal value to each category that you define using the RACF commands. The internal value is not displayed when the CATEGORY profile is listed using the RLIST command. If you use an application program to issue a RACROUTE REQUEST=DEFINE, ICHEINTY, or RACROUTE REQUEST=EXTRACT, TYPE=REPLACE macro that specifies internal CATEGORY values, and you use RRSF to keep your RACF databases synchronized, you must ensure that each CATEGORY has the same internal value assigned to it on each of the RACF databases. Use the RACF database unload utility (IRRDBU00) to unload the databases and check the CATEGORY profiles in the SECDATA class. If the internal category values are different, you must delete the category information from all user and resource profiles, delete the CATEGORY profiles, then redefine the

categories making sure that the command definitions occur in the same order on all systems. Once the CATEGORY profiles are identical on all systems, reassign the categories to users and resources. The SEARCH command with the CLIST option can be used to simplify this process.

Understanding Security Labels

You can use *security labels* to associate a specific security level with a set of (zero or more) security categories. Security labels, when associated with resources, users, and jobs, provide the following advantages over security levels and security categories:

- Security labels can be assigned to data that is not necessarily protected by a resource profile. For example, spool files are assigned the security label of their creators. In many cases, data that has been assigned a security label retains that security label from the time the data is created until the data is deleted. For example, when a spool file is created by a user or job that is running under a security label, the spool file is assigned the security label of the user or job. The spool file retains that security label until the spool file itself is deleted (which can be long after the user logs off or the job ends).
- Users can log on with different security labels at different times but with the same user ID; without security labels, a user always has the same (default) security level and categories.
- Output printed for a user or job by the Print Services Facility (PSF) can have a PSF identification label related to the security label of the user or job printed on every page.
- It is easier to maintain the security classification of users and data (changing the definition of a security label affects all users and resources that have that security label; you need not make the same change for many different profiles as you would for security levels and categories).

To know what system configuration you need in order to use security labels, see “B1 Configuration Requirements” on page 12. The following are some considerations related to security labels:

- If you assign a security label to a resource profile and activate the SECLABEL class, you must also assign a default security label to users who access the resource or resources protected by that profile. Further, the security label you assign to the users must allow the user to access the resource. For information on assigning security labels to users, see “Assigning Security Labels to Users” on page 104.
- If your installation uses the SETROPTS MLACTIVE option, all data protected by classes that require security labels must have security labels. For more information, see “Enforcing Multilevel Security (MLACTIVE Option)” on page 140.
- If your installation uses the SETROPTS MLS(FAILURES) option, there are tighter restrictions on attempted accesses to resources, depending on the kind of access attempt and the security labels of the resource and user. For more information, see “Security Label Authorization Checking” on page 647.
- If your installation uses the SETROPTS MLS(FAILURES) option, the first data set written to a tape volume defines the security label of any data set that is later written to the tape. For more information, see “Preventing the Copying of Data to a Lower Security Label (MLS Option)” on page 139.
- If your system includes products that do not support security labels when they invoke RACF, you should consider using the SETROPTS COMPATMODE option. See “Activating Compatibility Mode for Security Labels (COMPATMODE Option)” on page 139.

Creating a Security Label

To create a security label, you must create a profile in the SECLABEL class. The name of the profile is the security label.

Notes:

1. Any time you make a change to a SECLABEL profile, you must also refresh SETROPTS RACLIST processing for the SECLABEL class for the change to take effect. For example:

   ```
   SETROPTS RACLIST(SECLABEL) REFRESH
   ```

2. You need not activate the SECDATA class.

To create a SECLABEL profile, do the following:

1. Define the SECLEVEL profile to the SECDATA class using the RDEFINE command.

   ```
   RDEFINE SECDATA SECLEVEL UACC(NONE)
   ```

2. Define security levels as members of the SECLEVEL profile in the SECDATA class.

   ```
   RALTER SECDATA SECLEVEL ADDMEM(seclevel-name/seclevel-number ...)
   ```

3. Define the CATEGORY profile to the SECDATA class using the RDEFINE command.

   ```
   RDEFINE SECDATA CATEGORY UACC(NONE)
   ```

4. Define categories as members of the CATEGORY profile in the SECDATA class.

   ```
   RALTER SECDATA CATEGORY ADDMEM(category-1 category-2 ...)
   ```

5. For each security label, define a profile in the SECLABEL class. The profile names are the security labels available on your system, and must be no longer than eight characters. For each SECLABEL profile, specify a security level and (optionally) a set of categories. For example:

   ```
   RDEFINE SECLABEL security-label SECLEVEL(seclevel-name)
          ADDCATEGORY(category-1 category-2 ...)
   ```

6. Users cannot use a particular security label (for logging on, for submitting a job, or for specifying in a RACF profile) unless they have at least READ access authority to the SECLABEL profile of that name. For example, if the SECLABEL profile named EAGLE has UACC(NONE) specified, and you wanted user AHLEE and group GROUP1 to be able to log on with a security label of EAGLE, issue the following command:

   ```
   PERMIT EAGLE CLASS(SECLABEL) ACCESS(READ) ID(AHLEE GROUP1)
   ```

7. When you are ready to start using security labels, activate the SECLABEL class and activate SETROPTS RACLIST processing for the class. SETROPTS RACLIST processing is required for the SECLABEL class. You can do these two actions in one command:

   ```
   SETROPTS CLASSACT(SECLABEL) RACLIST(SECLABEL)
   ```

For more information on the commands used in this section, see *z/OS SecureWay Security Server RACF Command Language Reference*.

Relationship of SECLABEL to SECLEVEL and CATEGORY in Resource Profiles

If the SECLABEL class is active, any existing SECLEVEL and CATEGORY information in resource profiles is ignored. Even though SECLEVEL and CATEGORY information is ignored, you can maintain it (for example, using RACF commands). If the SECLABEL class is not active, RACF continues to use the SECLEVEL and CATEGORY function as it is currently implemented. Thus, by

activating or deactivating the SECLABEL class using the SETROPTS command, you choose to use or ignore security labels.

Security Label Naming Restrictions

Security Labels SYSHIGH, SYSLOW, and SYSNONE

SYSHIGH, SYSLOW, and SYSNONE are security labels that you can specify for resource and user profiles, but you cannot create them directly. (During RACF initialization, RACF creates SECLABEL profiles with these names if they do not already exist.)

- SYSHIGH combines the highest security level specified in the SECLEVEL profile with all of the categories defined in the CATEGORY profile.
- SYSLOW is the lowest security level specified in the SECLEVEL profile and *no* categories.
- SYSNONE is the same as SYSLOW, but is intended for use on resources that must be written to at different security labels when the SETROPTS MLS option is in effect (such as system catalogs).

These profiles do not actually contain the security levels and security categories that define SYSHIGH and SYSLOW. To find out what security levels and categories are used for security labels SYSHIGH and SYSLOW, enter:

```
RLIST SECDATA SECLEVEL
RLIST SECDATA CATEGORY
```

Figure 8 illustrates the SYSHIGH and SYSLOW security labels. The left side shows part of the RLIST output for the SECLEVEL profile. The right side shows part of the RLIST output for the CATEGORY profile. With the security levels and categories currently defined on the system, SYSHIGH includes security levels L200 and categories CAT1 through CAT5. SYSLOW includes only security level L10 (no categories).

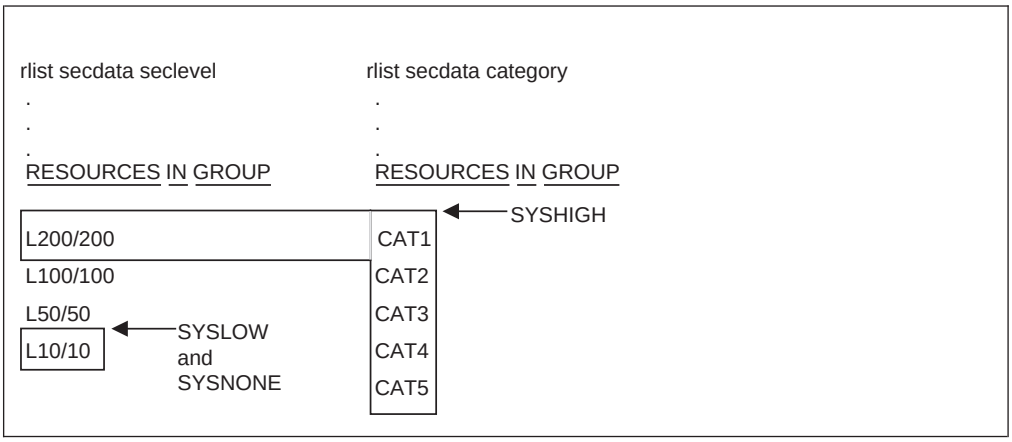


Figure 8. Sample SYSHIGH and SYSLOW Security Labels

The actual combination of security level and categories used to define the security labels SYSHIGH and SYSLOW is determined whenever the SETROPTS RACLIST command or SETROPTS RACLIST REFRESH command is issued for the SECLABEL class.

Assigning Security Labels to Users

A user profile can contain two security labels: a default security label, set by the security administrator, and the user's current security label, set when a user logs on using the TSO/E logon panel.

As security administrator, you should specify a default security label for each user that might need access to resources that are protected by security labels. To specify a user's default security label, enter the ADDUSER or ALTUSER command with the SECLABEL operand specified. You must also give each such user authority to use the security label. To give a user authority to use a security label, use the PERMIT command. For example:

```
ALTUSER userid SECLABEL(security-label)
PERMIT security-label CLASS(SECLABEL) ID(userid) ACCESS(READ)
```

Note: If you are adding security labels to a large number of user profiles, you can use the SEARCH command to generate a TSO CLIST that you can tailor (by editing) and then run. For example:

```
SEARCH CLASS(USER)
       CLIST('ALTUSER' 'SECLABEL(most-common-security-label)')
```

How Users Specify Current Security Labels

TSO users can override the default security label by specifying a value in the SECLABEL field on the logon panel or LOGON command. Batch users can also override the default security label by specifying the SECLABEL parameter on the JOB statement when a job is submitted to the system.

When a TSO user logs on using the TSO/E logon panel, and specifies a value in the SECLABEL field on the TSO/E logon panel, TSO records the value from the SECLABEL field in the TSO segment of the RACF user profile (if the TSO segment exists). The next time the user logs on, TSO displays the value from this field in the SECLABEL field on the logon panel as a default. If the user changes the SECLABEL field while logging on, TSO modifies the SECLABEL field in the user's TSO segment with the new current security label. This new value is used as the default that is presented for the next logon.

When you are migrating from security levels and security categories to security labels, you should consider setting the SECLABEL field using the ADDUSER and ALTUSER commands as follows:

```
ADDUSER userid SECLABEL(security-label)
ALTUSER userid SECLABEL(security-label)
```

Listing Security Labels

To display the security label stored in a resource profile, specify the ALL option on the LISTDSD and RLIST commands.

To display a user's *default* security label (the security label stored in the profile using the SECLABEL operand on the ADDUSER or ALTUSER command), issue the LISTUSER command with the user ID specified. For example:

```
LISTUSER JONES
```

Displaying the Current Security Label for a User ID

You cannot display another user's *current* security label. Only that user can display it. To display a user's current security label, ask the user to enter the LISTUSER command with no operands:

```
LISTUSER
```

When RACF displays a user's security label, RACF also displays the security level and any security categories that define it.

Finding Out Which Security Labels a User Can Use

To find out which security labels a user can specify, enter:

```
SEARCH CLASS(SECLABEL) USER(userid)
```

Note: Because SECLABELs must be RACLISTed, the class must be active before this SEARCH command can work.

Searching by Security Labels

To search for all of the profiles that have a particular security label, enter:

```
SEARCH CLASS(classname) SECLABEL(security-label)
```

For example:

```
SEARCH CLASS(TERMINAL) SECLABEL(EAGLE)
```

This command displays all of the terminal profiles that have security label EAGLE specified.

```
SEARCH CLASS(USER) SECLABEL(EAGLE)
```

This command displays all of the user profiles in which security label EAGLE is the default security label.

Notes:

1. You can search only one class at a time. If you do not specify a class, the DATASET class is searched by default.
2. RACF lists only the names of profiles to which the user has at least READ access authority.

Restricting Security Label Changes

You can restrict users who do not have the SPECIAL attribute from specifying security labels in resource profiles, or changing the definitions of security labels, by specifying the SECLABELCONTROL operand on the SETROPTS command. For more information, see "Restricting Changes to Security Labels (SECLABELCONTROL Option)" on page 139. When the SECLABELCONTROL option is off, any user can specify the SECLABEL operand if the user has at least READ access authority to the associated SECLABEL.

Requiring Security Labels

You can require that all work entering the system, including users logging on and batch jobs, have a security label assigned. For more information, see "Enforcing Multilevel Security (MLACTIVE Option)" on page 140. For a list of the products that must be installed, see "B1 Configuration Requirements" on page 12.

SECLABEL Tranquility Considerations

For an evaluated B1 system, when the security label of a resource is changed, none of the affected resources should be in use. The same thing applies when implicitly altering the definition of the SECLABEL profile by changing the SECLEVEL or CATEGORY profiles in the SECDATA class, or by changing which

security level or security categories apply to profiles in the SECLABEL class. For example, the following commands can implicitly change the definition of a security label:

```
RALTER SECDATA SECLEVEL ADDMEM(...)
RALTER SECLABEL EAGLE SECLEVEL(...)
RALTER SECLABEL EAGLE ADDCATEGORY(...)
RALTER SECLABEL EAGLE DELCATEGORY(...)
```

The inability of users to use any resources while either the users or resources security label is being changed is called *tranquility.*

For a list of the products that must be installed, see “B1 Configuration Requirements” on page 12.

Preventing Changes to Security Labels

You can prevent any user from changing the security label of a profile or from changing a SECLABEL profile. For more information, see “Preventing Changes to Security Labels (MLSTABLE Option)” on page 141.

Enforcing SECLABEL Tranquility

You can prevent users other than SPECIAL users, console operators, and started tasks from logging on, starting new jobs, or accessing resources. For more information, see “Quiescing RACF Activity (MLQUIET Option)” on page 137.

Any change of the security label in a data set profile is audited. In addition, if SETROPTS MLACTIVE is on, a type 83 SMF record is produced. This record contains a list of the cataloged data sets that are affected by the change, addition, or deletion of a data set profile.

- If the SETROPTS CATDSNS option is in effect, this list is a correct list of all data sets affected by the security label change.
- If the SETROPTS CATDSNS option is not on, the list of the affected data sets may not be complete. Thus, the SETROPTS CATDSNS option allows you to list and audit the data sets affected by the change in a particular profile.

Planning Considerations for Security Labels

Even though a single user can use more than one security label, it may be easier to assign each person a separate user ID for each security label used.

Also, some consideration should be taken before using resource profiles that, due to their naming structure, would either cause large numbers of resources to be protected by the same security label (for example: *userid.**) or protect resources that need to be accessed when the user is logged on at different security labels (for example, a NAMES data set).

These types of resources can be grouped into several categories and following certain procedures can minimize the impact of their use. You might consider creating data set profiles with the following naming structure:

```
userid.security-label.*
```

For example, if SMITH needs to use security labels EAGLE and THRUSH, create profiles like:

```
SMITH.EAGLE.* UACC(NONE) SECLABEL(EAGLE)
SMITH.THRUSH.* UACC(NONE) SECLABEL(THRUSH)
```

Some services create new data sets based on the user’s user ID. If the SETROPTS MLS option is in effect, whenever the user is using a security label that is different

from the security label that was in use when the data set was created. Applications that create such data sets should consider using the REXX RACVAR function package to determine the current security label for inclusion in the names of such data sets.

- **Read Only**

  In general, these resources pose no problem if they are protected by a profile with the lowest security label that is used. By being protected in this manner, they can be accessed any time the user is logged on. For example, system resources that are read by all users should be protected with the SYSLOW security label.

- **Read Mostly**

  These resources must also be protected by a profile at the lowest security label that is used. This allows the user to access them for read any time the user is logged on. If the resource needs to be updated (for example: new names being added to the nickname file '*userid*.NAMES.TEXT'), the user must log on at his or her lowest security label in order to update the file.

- **Read/Write but Able to Be Preallocated**

  Data sets such as the ISPF list and log data sets fall into this category. These data sets can be allocated from within a logon CLIST with a different data set used for each security label. It should be noted, however, that due to multiple data sets being used, updates to a data set at one security label would not cause updates to occur to a data set that is used for the same purpose at a different security label (for example, an SPF profile data set).

- **Data Sets Written to from Multiple Levels**

  An example of this type of data set is the SPF EDIT recovery data sets. The CLIST ISREDRTI that creates the ISPF Edit Recovery Table sets the default data set names for the recovery data sets to:

  ```
  '&ZUSER.&ZAPPLID&ZEROS&I.BACKUP'
  ```

  or for user ID RALPH to

  ```
  'RALPH.ISR0001.BACKUP'
  ```

  RACF has provided sample exits in SYS1.SAMPLIB that show how to solve this problem for ISPF and PDF data sets.

  A data set profile with a specific SECLABEL would cause a security label authorization failure when an attempt was made to write to the data set while logged on with a security label that was different from the one in the profile. Applications that create data sets in this manner can be used only if each user has access to only one security label.

When RACF checks a user's authority to use a terminal or console, or the authority of outbound work to use a JES writer, RACF uses "reverse MAC" (mandatory access checking). That is, the security label of the profile protecting the writer must be equal to or greater than the security label of the user or outbound work.

Security Labels: An Example

This example shows how to set up security labels to meet the following needs:

- Projects on the same system cannot access each other's data.
- Projects on the same system have access to common data (such as tools or data to which they need read, but not write, access)

In this example, there are two projects, A and B. Each project has one category (categories A and B). Although each project can have more than one category, for the purpose of this example, only one project is shown.

The system has four security levels:

REG Registered (currently the highest security level on the system)

RES Restricted

CON Confidential

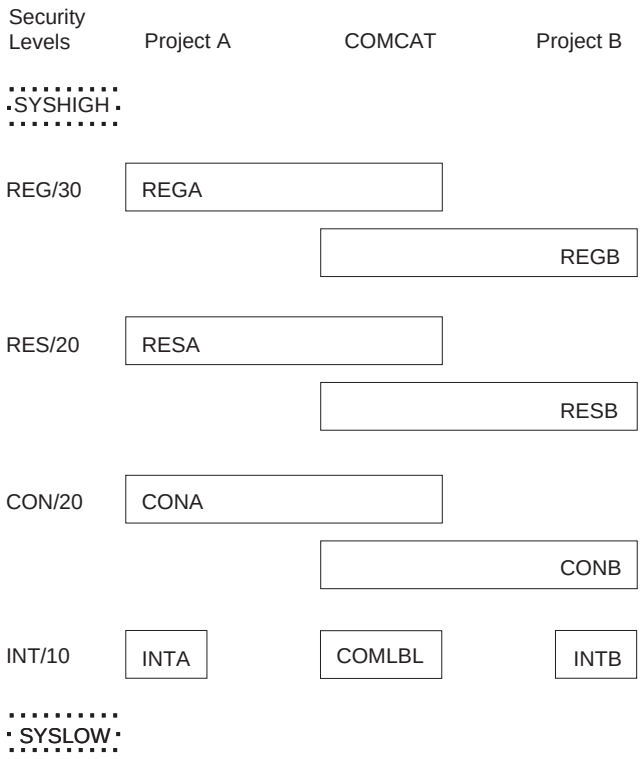
INT Internal

To give each of the two projects access to common data, do one of the following:

- If the SETROPTS MLACTIVE option is in effect, create a third security category (called COMCAT). When you create security labels, include the common category in security labels to allow users logged on at those security labels to read common data.

  Create a security label at the lowest defined security level (INT) that includes a third security category (COMCAT).
- If the SETROPTS MLACTIVE option is *not* in effect, assign no security label to profiles protecting the common data.

Data at the lowest security level (INT) within a particular project is shown primarily to show the effect of creating security labels that include no category for common data.



Notes:

1. SYSHIGH includes security level REG and categories A, B, and COMCAT.
2. SYSLOW includes security level INT and no categories.

The following commands assume that the security levels and categories have already been defined.

Commands to Define Security Labels for Project A:

```
RDEFINE SECLABEL REGA SECLEVEL(REG) ADDCAT(A COMCAT) UACC(NONE)
RDEFINE SECLABEL RESA SECLEVEL(RES) ADDCAT(A COMCAT) UACC(NONE)
RDEFINE SECLABEL CONA SECLEVEL(CON) ADDCAT(A COMCAT) UACC(NONE)
RDEFINE SECLABEL INTA SECLEVEL(INT) ADDCAT(A) UACC(NONE)
```

Commands to Define Security Labels for Project B:

```
RDEFINE SECLABEL REGB SECLEVEL(REG) ADDCAT(B COMCAT) UACC(NONE)
RDEFINE SECLABEL RESB SECLEVEL(RES) ADDCAT(B COMCAT) UACC(NONE)
RDEFINE SECLABEL CONB SECLEVEL(CON) ADDCAT(B COMCAT) UACC(NONE)
RDEFINE SECLABEL INTB SECLEVEL(INT) ADDCAT(B) UACC(NONE)
```

Command to Define the Common Security Label:

```
RDEFINE SECLABEL COMLBL SECLEVEL(INT) ADDCAT(COMCAT) UACC(NONE)
```

Commands to Assign Security Labels to Users:

Security administrator (a SPECIAL or group-SPECIAL user):

```
ALTUSER ADMIN SECLABEL(SYSHIGH)
```

A team leader has access to all security labels, but usually logs on to the highest security label in his or her project (REGA):

```
ALTUSER LEADERA SECLABEL(REGA)
PERMIT (REGA RESA CONA INTA) CLASS(SECLABEL) ID(LEADERA) ACCESS(READ)
```

A newly hired worker has access only to the lowest security label in his or her project:

```
ALTUSER WORKERA SECLABEL(INTA)
PERMIT INTA CLASS(SECLABEL) ID(WORKERA) ACCESS(READ)
```

Note: Similar commands are issued for project B users.

Commands to Assign Security Labels to Data for Project A:

```
ALTDSD 'GROUPA.REGA.**' SECLABEL(REGA)
ALTDSD 'GROUPA.RESA.**' SECLABEL(RESA)
ALTDSD 'GROUPA.CONA.**' SECLABEL(CONA)
ALTDSD 'GROUPA.INTA.**' SECLABEL(INTA)

RALTER VMMDISK GROUPA.294 SECLABEL(REGA)
RALTER VMMDISK GROUPA.293 SECLABEL(RESA)
RALTER VMMDISK GROUPA.292 SECLABEL(CONA)
RALTER VMMDISK GROUPA.291 SECLABEL(INTA)
```

Note: Similar commands are issued for project B data.

Commands to Assign Security Labels to Common Data:

```
ALTDSD 'COMMON.DATA' SECLABEL(COMLBL)
RALTER VMMDISK COMMON.191 SECLABEL(COMLBL)
```

What Users Can Do Based on Security Label Authorization Checking (SETROPTS NOMLS in Effect):

Note: Other authorization requirements, such as access lists and UACC, can prevent users from accessing data. This figure is limited to the controls

enforced by security label authorization checking (with the SECLABEL class active and the SETROPTS NOMLS option in effect).

Users who log on with project A security labels can only view data with security labels REGA, RESA, CONA, INTA, and COMLBL.

Users who log on with project B security labels can only view data with security labels REGB, RESB, CONB, INTB, and COMLBL.

Note that data with security label COMLBL can be seen by users in either project A or project B.

Users logged on with a particular security label:
- Can *read or update* data with lower security labels in that project, plus COMLBL
- Cannot read data with higher security labels in that project

For example, users logged on with security label RESA can:
- Read or update data with security label RESA, CONA, INTA, and COMLBL
- Cannot read data with security label REGA

Note: INTA represents something of a special case. Because the INTA security label is not defined to include the COMLBL category, users who log on with the INTA security label can only access data with the INTA security label.

What Users Can Do Based on Security Label Authorization Checking (SETROPTS MLS(FAILURES) in Effect):

Note: Other authorization requirements, such as access lists and UACC, can prevent users from accessing data. This figure is limited to the controls enforced by security label authorization checking when the SECLABEL class is active and the SETROPTS MLS(FAILURES) option is in effect.

Users who log on with project A security labels can only view data with security labels REGA, RESA, CONA, INTA, and COMLBL.

Users who log on with project B security labels can only view data with security labels REGB, RESB, CONB, INTB, and COMLBL.

Note that data with security label COMLBL can be seen by users in either project A or project B.

Users logged on with a particular security label:
- Can update data with that security label
- Can read data with lower security labels in that project, plus COMLBL
- Cannot read data with higher security labels in that project

For example, users logged on with security label RESA can:
- Update data with security label RESA
- Read data from lower security labels in that project: CONA, INTA, and COMLBL
- Cannot read data with security label REGA

Note: INTA represents something of a special case. Because the INTA security label is not defined to include the COMLBL category, users who log on with the INTA security label can only access data with the INTA security label.

Chapter 5. Specifying RACF Options

This chapter describes the RACF options you can specify to control how RACF operates on your system.

Using the SETROPTS Command

RACF provides many system-wide options for controlling the way it works on your system. You specify most of these options by issuing the SETROPTS command with the appropriate operands or filling in the appropriate ISPF panels.

This chapter discusses SETROPTS options that are useful to the RACF security administrator. It assumes that you have the SPECIAL attribute.

For a description of the SETROPTS options that are useful to the RACF auditor (with the AUDITOR attribute), see *z/OS SecureWay Security Server RACF Auditor's Guide*.

See *z/OS SecureWay Security Server RACF Command Language Reference* for a complete description of the SETROPTS command.

Hints for Selected SETROPTS Options

- If you are installing RACF for the first time, activate enhanced generic naming:

  ```
  SETROPTS EGN
  ```

- Do not issue the SETROPTS TERMINAL(NONE) command unless you have RACF-protected enough terminals so that users can log on. SETROPTS TERMINAL(NONE) prevents users from logging on to unprotected terminals.

  To recover from such a situation, submit a batch job that runs under a user ID with the SPECIAL attribute and that issues SETROPTS TERMINAL(READ).

- Some classes have a default return code of 8. If such a class is activated, but no profiles are defined, user activity that requires access in that class is prevented.

  Do not activate a class with a default return code of 8, either explicitly (by name) or implicitly (by means of a shared POSIT value), unless you have defined profiles for that class.

  RACF prevents you from accidentally activating all classes by misusing the SETROPTS CLASSACT(*) operand.

  If security labels have been assigned to resource profiles, do not activate the SECLABEL class by using SETROPTS CLASSACT(SECLABEL) unless you have assigned appropriate security labels to appropriate users.

  To recover from such a situation, log on as a user with the SPECIAL attribute, specifying SYSHIGH as the current security label. Then either assign security labels, or issue SETROPTS NOCLASSACT(SECLABEL).

- Do not issue the SETROPTS MLACTIVE(FAILURES) command unless you have assigned appropriate security labels to all users and to the resources that they must access.

  To recover from such a situation, log on as a user with the SPECIAL attribute, specifying SYSHIGH as the current security label. Then either assign security labels, or issue SETROPTS NOMLACTIVE.

SETROPTS Options for Initial Setup

You should consider specifying the options in this section when RACF is first installed. For a list of the options, see the chapter table of contents.

Establishing Password Syntax Rules (PASSWORD Option)

If you have the SPECIAL attribute, you can establish up to eight password syntax rules to verify that new passwords meet the installation standards. These rules allow you to control:

- The minimum and maximum length of passwords
- The character content of installation-selected positions in the passwords

You establish these rules by using the RULE*n* suboperand specified by the PASSWORD operand of the SETROPTS command. The following example shows how you can establish a syntax rule for new passwords for your installation.

```
SETROPTS PASSWORD(RULE1(LENGTH(8) VOWEL(1,3,5:8) NUMERIC(2,4)))
```

The command establishes syntax rule RULE1. Syntax rule RULE1 specifies that new passwords must be 8 characters in length, must contain vowels in positions 1, 3, 5, 6, 7, and 8, and must contain numbers in positions 2 and 4. Thus, the password A2E2EAEE follows the rule, and C3DMIER5 does not.

If you do not define a value for every position specified by the LENGTH value, the undefined positions can contain any combination of alphanumeric characters.

Note: If the RACF ISPF panels are installed, you should consider using the RACF ISPF panels to set up password syntax rules.

Setting the Maximum Password Change Interval (PASSWORD Option)

The INTERVAL suboperand specifies the system default for the number of days that a user's password is to remain valid. The following example specifies that each user's password remain valid for 60 days (as long as the system default for these users remains 60 days):

```
SETROPTS PASSWORD(INTERVAL(60))
```

This value becomes effective immediately as:

- A default value for new users whom you define to RACF through the ADDUSER command
- An upper limit for users who specify the INTERVAL operand on the PASSWORD command

When users are defined to RACF and have access to the system, they can use the INTERVAL operand of the PASSWORD command to set their own change interval to a value less than 30 or to a value less than that which you specified on the INTERVAL operand of the SETROPTS command (if you did so).

Note: When you invoke this option, the change interval values in existing user profiles are not modified. RACF uses the smaller of the two values.

The initial system default for the change interval is 30 days.

Extending Password and User ID Processing (PASSWORD Option)

If you have the SPECIAL attribute, you can specify the WARNING/NOWARNING, HISTORY/NOHISTORY, REVOKE/NOREVOKE, and INTERVAL options.

Use the PASSWORD option on the SETROPTS command to provide the following functions:

- The WARNING suboperand enables you to specify when RACF should issue a password expiration message. If you specify WARNING, RACF issues a message each time a user logs on to TSO or submits a batch job with a password a specified number of days before the password expires. The following example specifies that RACF issue a warning message 5 days before a password expires:

  ```
  SETROPTS PASSWORD(WARNING(5))
  ```

  If NOWARNING is in effect, RACF does not issue a warning message before a password expires.
- The HISTORY suboperand enables you to specify the number of previous passwords that RACF saves and compares with an intended new password. If there is a match with one of these previous passwords, or with the current password, RACF rejects the intended new password. If you increase the

password HISTORY number, RACF saves and compares that number of passwords to the new password. If you reduce the password HISTORY number, passwords in the user profile that are beyond the newly specified HISTORY number are never deleted and continue to be used for comparison. For example, if the HISTORY number is 12 and you reduce that HISTORY number to 8, RACF continues to compare the old passwords 9 through 12 with the intended new password.

The following example specifies that RACF save and compare 10 previous passwords, in addition to the current password, with an intended new password:

```
SETROPTS PASSWORD(HISTORY(10))
```

Note: When the user attempts to change his or her own password, RACF verifies that the intended new password does not match a previous entry in the history list, saves the current password in the list, and changes the password to the new value. NOHISTORY specifies that the new password information is compared only to the current password. Any previous password history information is neither deleted nor changed.

- REVOKE enables you to specify how many consecutive password verification attempts RACF is to permit before it revokes a user ID on the next attempt. Protected user IDs are not revoked based on consecutive incorrect password attempts. See “Defining Protected User IDs” on page 86 for more information.

  The following example specifies that RACF is to allow 4 consecutive incorrect password attempts. A fifth incorrect password attempt revokes the user ID:

  ```
  SETROPTS PASSWORD(REVOKE(4))
  ```

  After RACF revokes the user ID, you can activate the user ID with the RESUME operand of the ALTUSER command if you have the SPECIAL or group-SPECIAL attribute or are the owner of the profile. If NOREVOKE is in effect, consecutive incorrect passwords are ignored.

Revoking Unused User IDs (INACTIVE Option)

The INACTIVE operand of the SETROPTS command causes RACF to revoke the user’s right to use the system if the user ID has remained unused beyond a specified number of days. RACF revokes the user the next time the user attempts to enter the system.

The following example specifies that RACF revoke a user ID if it is unused for over 30 days:

```
SETROPTS INACTIVE(30)
```

Notes:

1. New users who never use the system are not revoked because of inactivity. This is true unless an administrator has used the ALTUSER or PASSWORD commands to set their password or someone makes an unsuccessful attempt to log on.
2. If you issue the SETROPTS INACTIVE(30) command and a user has not done any of the following in 31 days:
   - Logged on
   - Submitted a job
   - Changed their password by any method
   - Attempted an unsuccessful logon
   - Received a directed command or output from RACF

that user is considered revoked. However, the user is not actually revoked and the output of the LISTUSER command does not show that the user is revoked until the user next attempts to log on or submit a job.

3. When you allow the user to start using the system again (using the RESUME operand on the ALTUSER command), RACF resets the effective date with which the period of inactivity starts.

If NOINACTIVE is in effect, RACF does not check the user ID against an unused user ID interval.

If NOINITSTATS is in effect, the INACTIVE, REVOKE, HISTORY, and WARNING options cannot be used.

Activating List-of-Groups Checking (GRPLIST Option)

List-of-groups authority checking supplements the normal RACF access authority checking by allowing all groups of which a user ID is a member to enter into the access list checking process. This process replaces the checking that compares the current connect group with the resource's access list, and can expand a user's ability to access resources. If list-of-groups checking is active, then regardless of which group the user is logged on to, RACF recognizes the user's group-related authorities in other connect groups. If a user is in more than one group and tries to access a resource, RACF uses the highest authority allowed by the user's list of groups and the resource's access list.

Note: A user's current connect group is the group entered on the logon panel or with the LOGON command. If no group is specified at logon, the user's default group is used.

For example, the user is logged on to Group B (the current connect group) and tries to access a resource. The resource's access list does not contain the user's user ID or the group name for Group B, but it does contain the group name for Group A with an associated access authority of READ. If the user is a member of Group A (and Group B) and list-of-groups checking is active, the user can access the resource, even though the user is logged on to Group B. (This example assumes that other RACF checks, such as security classification checking, are met.)

Similarly, if list-of-groups checking is active, RACF recognizes the user's group-related attributes (such as group-SPECIAL) in other connect groups, regardless of which group the user is logged on to. However, the user still has each group-related attribute only within the scope of that group in which the user is assigned the attribute. (For more information on the scope of a group, see "Chapter 3. Defining Groups and Users" on page 49.)

For example, in Figure 6 on page 83, say USER1 is also connected to GROUP3, but without group-SPECIAL for GROUP3. If list-of-groups checking is not active and USER1 logs on to GROUP3, RACF does not recognize that USER1 has group-SPECIAL authority to GROUP1 resources.

If list-of-groups checking is active and USER1 logs on to GROUP3, USER1 has group-SPECIAL authority to GROUP1 resources. However, USER1 does not have group-SPECIAL authority to GROUP3 resources. Likewise, if list-of-groups checking is active and USER1 logs on to GROUP1, USER1 has group-SPECIAL authority to GROUP1 resources, but not GROUP3 resources.

If you have the SPECIAL attribute, you can specify list-of-groups checking by using the GRPLIST option of the SETROPTS command as shown in the following example:

```
SETROPTS GRPLIST
```

To use current connect group checking, specify the NOGRPLIST option on the SETROPTS command.

IBM recommends the use of the GRPLIST option because it eases administration and minimizes the number of times the user might have to log off and log back on to access resources.

GRPLIST Considerations for z/OS UNIX

z/OS UNIX groups are RACF groups that have an z/OS UNIX group identifier (GID) defined in the OMVS segment of the group's profile. Authority checks for access to hierarchical file system (HFS) files and directories use the GID in the user's current connect group and up to 300 supplementary groups (if SETROPTS GRPLIST is active) to make group access decisions. The limit of 300 supplementary groups is the same limit defined by the `NGROUPS_MAX` variable of the POSIX standard. Authority checks for other system resources use the RACF current group and list-of-groups support.

For more information, see "Defining Group Identifiers (GIDs)" on page 504.

Setting the RVARY Passwords (RVARYPW Option)

If you have the SPECIAL attribute, you can specify passwords that the operator must use to respond to RVARY command requests to:

- Switch the RACF databases
- Change RACF status (ACTIVATE or DEACTIVATE)
- Change mode (DATASHARE or NODATASHARE)

You can specify the passwords using the RVARYPW operand on the SETROPTS command. RACF allows you to specify separate passwords for switching the databases and for changing RACF status. The following example specifies HAPPY as the switch password and RABBIT as the status password:

```
SETROPTS RVARYPW(SWITCH(HAPPY) STATUS(RABBIT))
```

Password names must conform to the following rules:

- Passwords can be up to 8 characters in length.
- Valid characters for names are alphabetic (A through Z), numeric (0 through 9), and national (#, @, and $).

When RACF is first initialized, the switch password and the status password are both set to YES.

Restricting the Creation of General Resource Profiles (GENERICOWNER Option)

If you have the SPECIAL attribute, you can restrict the creation of profiles in general resource classes. To do this:

1. Issue a SETROPTS GENERICOWNER command.
2. Define a ** profile for the class, with yourself as owner. (This prevents users lacking special authority from being able to define profiles in the class.)

3. Define a *top* profile for each user, covering the subset of resources in the class which the user is allowed to create. Each user should be the owner of this *top* profile.

You have created an environment where the user can create only profiles that are *more specific* than the user's *top* profile. The only other users who can create profiles in the user's subset of the class are:

- A user with SPECIAL authority
- A user who has group-SPECIAL authority over a user who owns the *top* profile

For example, assume that neither JOE nor RONN have the SPECIAL or group-SPECIAL attribute. If the GENERICOWNER option is in effect, and user RONN is the owner of a JESSPOOL profile called NODEA.RONN.**, JOE cannot create profile NODEA.RONN.DATA.**, even though JOE has the CLAUTH(JESSPOOL) attribute.

Note: The GENERICOWNER operand does not affect the DATASET class. It cannot be activated for individual classes. When active, GENERICOWNER affects all general resource classes except the PROGRAM class and general resource grouping classes.

For example, when working with general resource grouping classes, assume that profile A* exists in the TERMINAL class and is owned by a group that user ELAINE does not have group-SPECIAL authority to. If the GENERICOWNER option is in effect, it will prevent user ELAINE from defining a more specific profile in the member class (for example, by using the command `RDEF TERMINAL AA*`). However, having the GENERICOWNER option in effect will **not** prevent user ELAINE from defining a profile if specified on the ADDMEM operand for the grouping class profile (such as with the command `RDEF GTERMINL profile-name ADDMEM(AA*)`).

You can alternatively choose to make a group the owner of the *top* profile for a given subset in the class. In this case, only a user with group-SPECIAL authority for the group, or with SPECIAL authority, can create profiles in the subset.

The *top* profile must end in a single asterisk (*), double asterisks (**), or one or more percent signs (%). More specific profiles are profiles that match the less specific *top* profile name character for character, up to the ending asterisks or percent signs in the less specific name.

In a search for the less specific profile, a match is found if *all* of the following are true:

- The profile name ends in a single asterisk (*), double asterisks (**), or one or more percent signs (%).
- All characters preceding the asterisks or percent signs (* or ** or %) match the corresponding characters in the resource name exactly.
- The characters matching the percent signs (%) in the less-specific profile are not an asterisk (*) or period (.) in the resource name. The length of the profile must be the same for this case.

For example, to allow USERX to RDEFINE A.B in the JESSPOOL class, you need profile A.* in the JESSPOOL class, which is owned by USERX.

To cancel this option, specify NOGENERICOWNER on the SETROPTS command.

Attention

Issuing SETROPTS GENERICOWNER can prevent users with the CLAUTH attribute in general resource classes from creating profiles as they are accustomed to. Therefore, make these users OWNER of appropriate "top" generic profiles in the class. For an example, see "Delegating Authority to Profiles in the FACILITY Class" on page 218.

Activating General Resource Classes (CLASSACT Option)

If you have the SPECIAL attribute, you can specify that RACF provides access authorization checking for general resource classes. You can specify this option for selected general resource classes with the CLASSACT operand of the SETROPTS command.

The following example shows how to specify RACF access authorization checking for the TERMINAL and CONSOLE resource classes.

```
SETROPTS CLASSACT(TERMINAL CONSOLE)
```

IBM does not recommend that you activate all RACF classes. You should activate only the classes that are important to your installation, as some classes have a default return code of 8. Those classes should only be activated after you have defined the necessary profiles to allow access to resources. RACF prevents you from accidentally activating all classes using the SETROPTS CLASSACT(*) operand.

For information on activating protection for specific general resource classes, check the index of this book for the class name.

If you have the SPECIAL attribute, you can also specify the NOCLASSACT operand on the SETROPTS command. This operand indicates that RACF performs no access authorization checking for selected general resource classes. If you specify NOCLASSACT(*), RACF does not perform access authorization checking for any of the classes in the class descriptor table (CDT). However, you can still define resource profiles to RACF through the RDEFINE command.

Special Considerations for Auditing z/OS UNIX

The following classes are defined only for auditing z/OS UNIX security events and are *not* used for authorization checking:

- DIRACC
- DIRSRCH
- FSOBJ
- FSSEC
- IPCOBJ
- PROCACT
- PROCESS

No profiles can be defined in these classes. They are used to define the auditing options for z/OS UNIX security events. The classes do not need to be active to control auditing.

SETROPTS LOGOPTIONS can be used to specify logging options for all of the classes. Additionally, SETROPTS AUDIT options are used to control auditing of

some events for the FSOBJ and the PROCESS classes. For more information on the SETROPTS command, see *z/OS SecureWay Security Server RACF Command Language Reference*.

Activating Generic Profile Checking and Generic Command Processing

If you have the SPECIAL attribute, you can activate or deactivate generic profile checking either on a class-by-class basis or for all classes. You can specify this option with the GENERIC and NOGENERIC operands of the SETROPTS command. The following example shows how to activate generic profile checking for the DATASET class.

```
SETROPTS GENERIC(DATASET)
```

If you specify GENERIC(*), you activate generic profile checking for the DATASET class and all classes in the class descriptor table (CDT) except resource group classes (such as GTERMINL and GDASDVOL).

If you want to perform maintenance on the generic profiles in the RACF database, you may want to temporarily deactivate generic profile checking but allow RACF command processors to update generic profiles. You can specify this environment with the NOGENERIC and GENCMD operands of the SETROPTS command. The following example shows how to specify this environment for the DATASET class.

```
SETROPTS NOGENERIC(DATASET) GENCMD(DATASET)
```

NOGENERIC and NOGENCMD are in effect when a RACF database is first initialized using IRRMIN00.

Note: IBM recommends the use of generic profiles when you can use a generic profile to protect multiple resources and thus ease the administration. You should consider issuing SETROPTS GENERIC(*) so that generic profiles are usable in all classes.

If there is a global access checking table entry of $RACUID.**/ALTER for data sets, users can create unprotected data sets even if protect-all is in effect. However, *other users* are not allowed to access those data sets.

Activating Statistics Collection (STATISTICS Option)

Using the SETROPTS STATISTICS option does the following:

- RACF maintains two sets of statistics in a discrete resource profile. One set counts all activity for the resource or profile. The other set counts activity for each entry in the access list. It can be difficult to compare the two sets of statistics meaningfully, unless you understand how RACF maintains the statistics. For more information, see the “STATISTICS Example”.
- If a specific resource has unique security concerns, you should protect it with a discrete profile.

  To see how that resource is being accessed and how many times it is being accessed, you can initiate STATISTICS. Remember that the initiation of STATISTICS is *system-wide* for all discrete profiles within a particular resource class across your system. Depending on the number of discrete profiles in the various resource classes, turning on STATISTICS may negatively affect performance.

STATISTICS Example

To help you understand how RACF maintains statistics, consider the following:

- USER1.DATA is a data set profile.

- USER1.DATA has a universal access (UACC) of READ.
- USER2 is in the access list with READ authority.
- USER3 is in the access list with UPDATE authority.
- GROUP1 is in the access list with READ authority.
- GROUP2 is in the access list with UPDATE authority.
- USER4 belongs to both groups, GROUP1 and GROUP2.
- There is no entry for &RACUID.* in the global access checking table.

If USER1 reads USER1.DATA, the overall READ count in the profile increases by one. No counts in the access list are changed, because access lists are not used when users process their own data.

If USER2 reads the data set, two counts are updated: the overall READ count and the count in USER2's access list entry.

If USER3 reads the data set, two counts are updated: the overall READ count and the count in USER3's access list entry (even though the entry says UPDATE). The counts in the access list merely record that access was granted by that entry. The access granted can be as specified by the entry, or a lower level, as in this example.

If list-of-groups processing is active (through SETROPTS GRPLIST) and USER4 reads the data set, RACF examines the access list to see if any of USER4's groups are in the list. If any of the groups is found, the entry with the highest authority is used. In this case, the access list entry for GROUP2 (UPDATE) increases, along with the overall READ count for the profile.

If any other user or group reads the data set, it gains access because of the universal access of READ, and the overall READ count increases. If any user with OPERATIONS authority updates the data set, the overall UPDATE count increases.

Using Options in RACROUTE REQUEST=VERIFY Statistics Collection

A user with a SPECIAL attribute can request RACF to record statistics during the REQUEST=VERIFY processing. REQUEST=VERIFY is issued when a user is logging on to a system or a batch job is entering a system as well as when RACF does such work as a directed command, application update, or password change on behalf of the user. The statistics RACF maintains are:

- The date and time REQUEST=VERIFY is issued for a particular user
- The number of REQUEST=VERIFYs for a user to a particular group
- The date and time of the last REQUEST=VERIFY for a user to a particular group

The statistics must be maintained if you intend to use the INACTIVE, REVOKE, HISTORY, and WARNING options.

If, for your system, you do not need all of the statistics, you do not need to use the above four options, and you have the SPECIAL attribute, you can issue SETROPTS NOINITSTATS which reduces the RACF database I/O associated with a REQUEST=VERIFY function.

If NOINITSTATS is in effect, the INACTIVE, REVOKE, HISTORY, and WARNING options cannot be used.

If you have the SPECIAL attribute, you can also specify the INITSTATS operand on the SETROPTS command to indicate that you want RACF to record REQUEST=VERIFY statistics, as shown in the following example:

```
SETROPTS INITSTATS
```

INITSTATS is in effect when RACF is first initialized.

Note: Being enabled for sysplex communication may reduce the overhead associated with INITSTATS.

Bypassing Resource Statistics Collection

If you have the SPECIAL attribute, you can request that RACF bypass the recording of statistical information in discrete profiles for the DATASET class for classes defined in the class descriptor table (CDT). You specify this option with the NOSTATISTICS operand of the SETROPTS command.

The statistics you can bypass include:

- The date that the resource was last referenced
- The date that the resource was last updated (not recorded for terminals)
- The number of times that the resource was accessed for each of the following access authorities: ALTER, CONTROL, UPDATE, and READ (only READ count is recorded for terminals)
- The number of times that each user or group in the access list has accessed the resource

If you have the SPECIAL attribute, you can also specify the STATISTICS operand on the SETROPTS command and identify the classes for which you want RACF to record statistical information.

If you specify an asterisk (*), you activate the recording of statistical information for all resource classes.

When a RACF database is first initialized using IRRMIN00, STATISTICS is in effect for the DATASET, DASDVOL, TAPEVOL, and TERMINAL classes. Because statistics recording affects system performance, IBM recommends that you deactivate this option until your installation evaluates the need to use it against its potential performance impact. See *z/OS SecureWay Security Server RACF System Programmer's Guide* for more information.

Activating Global Access Checking (GLOBAL Option)

If you have the SPECIAL attribute, you can activate or deactivate global access checking on a class-by-class basis or for all classes. You can specify this option with the GLOBAL and NOGLOBAL operands of the SETROPTS command. The following example shows how to activate global access checking for the FACILITY class.

```
SETROPTS GLOBAL(FACILITY)
```

If you specify GLOBAL(*), you activate global access checking for the DATASET class and all classes in the class descriptor table (CDT) except group resource classes.

When you use the SETROPTS command to activate (or reactivate) global access checking for a class, RACF builds (or updates) the in-storage global access checking tables. However, you can use the RDEFINE and RALTER commands to maintain profiles on the database, regardless of whether the global access checking option is active for a class.

NOGLOBAL is in effect when RACF is first initialized.

Note: The SETROPTS GLOBAL(*classname*) command is propagated when the system is enabled for sysplex communication.

RACF-Protecting All Data Sets (PROTECTALL Option)

If you have the SPECIAL attribute, you can activate protect-all processing by using the PROTECTALL operand of the SETROPTS command. If protect-all is active, a user can create or access a data set only if the data set is RACF-protected by either a discrete or generic profile, *or* the access is allowed by global access checking. Note that if protect-all is in effect, generic profile checking should also be in effect for the DATASET class. Otherwise, users can create only data sets that are protected by discrete profiles. The following examples show how to specify these options:

```
SETROPTS PROTECTALL
SETROPTS GENERIC(DATASET)
```

Notes:

1. PROTECTALL requires that you RACF-protect all data sets. This protection includes tape data sets if your installation specifies TAPEDSN on the SETROPTS command.
2. After defining, altering, or deleting a generic profile, the following command ensures that the profile is in effect during authorization checking:

   ```
   SETROPTS GENERIC(DATASET) REFRESH
   ```
3. Started procedures with the privileged or trusted attribute and users with the SPECIAL attribute can access a data set that has no RACF profile, even if protect-all is in effect. These exceptions allow recovery if a critical profile is accidentally deleted.
4. If there is a global access checking table entry of &RACUID.**/ALTER for data sets, users can create unprotected data sets even if protect-all is in effect. However, *other users* cannot access those data sets.

Protect-all also has a warning option that allows the request even though the data set is not protected, but sends a warning message to the user and the MVS console. For example:

```
SETROPTS PROTECTALL(WARNING)
```

Note: Before using PROTECTALL(WARNING), you should consider taking the following steps to reduce the number of messages generated:

- Ensure that a RACF user or group profile is defined for all catalog aliases.
- Ensure that all RACF users and groups have a generic data set profile of the form:

  '*high-level-qualifier*.*'

  or, if SETROPTS EGN is in effect:

  '*high-level-qualifier*.**'

Protect-all applies to all data sets that do not have system-generated temporary names and that do not have names that begin with **SYSUT. You can extend protect-all to include temporary data sets with system-generated names by using the naming conventions table to modify the name that RACF uses to look like a permanent name. If your installation uses nonstandard names for temporary data sets, you must also predefine entries in the global access checking table that allow these data sets to be created and accessed.

RACF Options

If you have the SPECIAL attribute, you can also deactivate protect-all processing by using the NOPROTECTALL operand.

NOPROTECTALL is in effect when RACF is first initialized.

Activating JES2 or JES3 RACF Support

You have several choices about how RACF supports JES. Table 8 tells you where to find more information about RACF support on JES:

Table 8. RACF Support on JES

RACF Option	Go to section
SETROPTS JES(BATCHALLRACF)	“Forcing Batch Users to Identify Themselves to RACF” on page 440
SETROPTS JES(XBMALLRACF)	“Support for Execution Batch Monitor (XBM) (JES2 Only)” on page 440
SETROPTS JES(EARLYVERIFY)	“JES User ID Early Verification” on page 441
SETROPTS JES(NJEUSER)	“Understanding Default User IDs” on page 467
SETROPTS JES(UNDEFINEDUSER)	“Understanding Default User IDs” on page 467

Preventing Access to Uncataloged Data Sets (CATDSNS Option)

If you have the SPECIAL attribute, you can use the CATDSNS operand of the SETROPTS command to keep users who do not have the SPECIAL attribute from gaining access to data sets that DFP controls. These data sets include system temporary data sets and data sets that are not cataloged. Users cannot read from tape data sets, and they cannot read from or write to DASD data sets.

There are some exceptions. For example, CATDSNS processing does not fail the request:

- If the user has READ access to a resource called ICHUNCAT.*dataset-name* in the FACILITY class
- If the data set is protected by a discrete profile
- If the data set is created and used within the same job or TSO session. If this data set is not cataloged before job termination or TSO logoff, it is not accessible to any other job or TSO session.

To activate the CATDSNS option, enter:

```
SETROPTS CATDSNS
```

The CATDSNS option prevents users from using STEPCAT or JOBCAT statements. An installation that wants to allow its users to use those functions can override the restriction by granting users READ access to the ICHUSERCAT resource in the FACILITY class.

In addition, if SETROPTS MLACTIVE is in effect, RACF produces one or more type 83 SMF records whenever a data set profile is added, changed, or deleted. This record contains the list of the cataloged data sets that are affected by the change, addition, or deletion of the profile. To see what products you need for this function, see “B1 Configuration Requirements” on page 12.

If the SETROPTS CATDSNS option is not in effect, the list of the affected data sets may not be complete. Therefore, using the SETROPTS CATDSNS option allows you to list and audit the data sets affected by the change in a particular profile.

Because the SETROPTS CATDSNS option prevents users without the SPECIAL attribute from accessing uncataloged data sets (except as noted above), the list of data set names provided by the DSNS operand on the LISTDSD command is identical to the list of all data sets affected by the profile change.

You can also specify CATDSNS(WARNING), which allows accesses, but sends a warning message to the user and the security administrator.

To cancel the CATDSNS option, specify NOCATDSNS on the SETROPTS command.

Activating Enhanced Generic Naming for the DATASET Class (EGN Option)

If you have the SPECIAL attribute, you can activate enhanced generic naming for the DATASET class by issuing the SETROPTS command with the EGN operand:

```
SETROPTS EGN
```

When you activate this option, RACF allows you to specify the generic character ** (in addition to the generic characters * and %) when you define any of the following:

- A generic profile in the DATASET class
- An entry in the global access checking table for the DATASET class

Note that enhanced generic naming changes the meaning of the generic character * for generic data set profiles. (SETROPTS EGN has no effect on general resource profiles.) In addition, a generic profile such as *hlq*.* defined while SETROPTS NOEGN is in effect, will appear as *hlq*.*.** when listed after EGN is activated.

For information on specifying profile names with enhanced generic naming, see *z/OS SecureWay Security Server RACF Command Language Reference*.

To deactivate enhanced generic naming for the DATASET class, issue the SETROPTS command with the NOEGN operand.

Note: IBM strongly recommends that you *not* deactivate enhanced generic naming after data set profiles have been created while enhanced generic naming was active.

NOEGN is in effect when a RACF database is first initialized using IRRMIN00.

Controlling Data Set Modeling (MODEL Option)

The MODEL operand of the SETROPTS command allows you to automatically supplement the information that is normally placed in new data set profiles by ADSP, PROTECT=YES, or ADDSD. Modeling can be effective for user, group, and GDG data sets on an individual user ID or group name basis. You control this processing with the MODEL(USER), MODEL(GROUP), and MODEL(GDG) operands of the SETROPTS command. The following example shows how to specify this option for USER data sets:

```
SETROPTS MODEL(USER)
```

To specify this option, you must have the SPECIAL attribute.

Note: The FROM(*profile-name*) operand on the ADDSD command overrides any specifications from the MODEL(USER) or MODEL(GROUP) operands.

If you specify MODEL(NOGDG), MODEL(NOUSER), or MODEL(NOGROUP), RACF does not use a model data set profile for new GDG, USER, or GROUP data sets, respectively. For more information, see “Automatic Profile Modeling for Data Sets” on page 163.

If you specify NOMODEL, RACF does not use automatic model processing for GDG, group, or user data sets.

NOMODEL is in effect when a RACF database is first initialized using IRRMIN00.

Bypassing Automatic Data Set Protection (NOADSP Option)

If you have the SPECIAL attribute, you can specify that RACF ignore the ADSP attribute (if specified in user profiles). To do this, enter:

```
SETROPTS NOADSP
```

With the SETROPTS NOADSP operand in effect, RACF does not automatically create discrete data set profiles when users who have the ADSP attribute create new data sets. IBM recommends the NOADSP option because it reduces the number of data set profiles in the RACF database. Using generic data set profiles is generally more efficient.

You can reinstate normal ADSP processing with the ADSP operand.

ADSP is in effect when a RACF database is first initialized using IRRMIN00.

Displaying and Logging Real Data Set Names (REALDSN Option)

If your installation is using the naming conventions table or installation exits to convert data set names, you can specify that RACF put the actual data set names used by RACROUTE REQUEST=AUTH and REQUEST=DEFINE into any SMF log record and operator messages. To do this, enter:

```
SETROPTS REALDSN
```

Putting the REALDSN option into effect ensures that log printouts and operator messages identify data sets by their real names rather than by the data set names that are created by installation exit routines to conform to RACF naming conventions. To specify this option, you must have the SPECIAL attribute.

Note: This option has no effect on single-qualifier data set names (unless they have been modified by the naming conventions table or an exit routine), whose real data set names continue to be the prefixed ones. For more information, see *z/OS SecureWay Security Server RACF System Programmer’s Guide*.

NOREALDSN is in effect when a RACF database is first initialized using IRRMIN00.

Protecting Data Sets with Single-Qualifier Names (PREFIX Option)

You can RACF-protect data sets that have names consisting of only a single qualifier (that is, single-level names). To get RACF protection for single-qualifier names, issue the SETROPTS command with the PREFIX operand to activate the facility and define a prefix. If you use the SETROPTS command with the PREFIX operand to define a prefix (high-level qualifier), RACF internally modifies

single-qualifier names by adding the high-level qualifier when it processes requests for the data set. The prefix must be an existing group name and cannot be the name used as the high-level qualifier of any actual data sets or data set profiles in the system. The following example shows how to RACF-protect data sets with single-qualifier names with the prefix RAC1LVL:

```
SETROPTS PREFIX(RAC1LVL)
```

Attention

If you do not issue the SETROPTS command with the PREFIX operand, a system ABEND occurs if a discrete profile is created when a user tries to create a data set with a single-qualifier name.

To specify the PREFIX or NOPREFIX operands, you must have the SPECIAL attribute.

Activating Tape Data Set Protection (TAPEDSN Option)

If you have the SPECIAL attribute, you can activate tape data set protection by using the TAPEDSN operand of the SETROPTS command. When you activate tape data set protection, RACF refers to profiles in the DATASET class when verifying a user's access authority to a tape data set. The following example shows how to specify this option:

```
SETROPTS TAPEDSN
```

If you have the SPECIAL attribute, you can also deactivate tape data set protection by using the NOTAPEDSN operand on the SETROPTS command.

NOTAPEDSN is in effect when a RACF database is first initialized using IRRMIN00.

Activating Tape Volume Protection (CLASSACT(TAPEVOL) Option)

If you have the SPECIAL attribute, you can activate tape volume protection by using the CLASSACT(TAPEVOL) operand of the SETROPTS command. When you activate tape volume protection, RACF refers to profiles in the TAPEVOL class when verifying a user's access authority to a tape volume. If both the TAPEVOL class and TAPEDSN are active, RACF maintains profiles in both the TAPEVOL and DATASET classes. Data fields within these two profiles (data set name in the TAPEVOL profile and volume serial in a discrete data set profile) link the two profiles to each other. The following example shows how to activate tape volume protection:

```
SETROPTS CLASSACT(TAPEVOL)
```

If you have the SPECIAL attribute, you can also deactivate tape volume protection by using the NOCLASSACT(TAPEVOL) operand on the SETROPTS command.

Note: If your installation has a tape management system, you might consider running with TAPEDSN active and TAPEVOL inactive. In this case, you maintain tape volume security because the tape management system controls access to volumes. For more information, see "Choosing Which Tape-Related Options to Use" on page 175.

Establishing a Security Retention Period for Tape Data Sets (RETPD Option)

The RACF security retention period is the number of days that RACF protection remains in effect for a tape data set. For example, to select tape volumes to return to the scratch pool, a tape librarian can issue the SEARCH command with the EXPIRES operand. When the librarian issues this command, RACF uses the security retention period to check if RACF protection for all data sets on a tape volume has expired. If RACF protection has expired, the tape volume can be returned to the scratch pool.

If you define a tape volume with a TVTOC, RACF uses the security retention period when checking the authority to overwrite a data set on the volume with a data set of a different name. Before opening the tape data set for output, RACF ensures that the security retention periods for all of the following data sets on the volume have expired.

Users can specify a security retention period on the ADDSD and ALTDSD commands, or, for data sets covered by a discrete profile, by the use of the EXPDT/RETPD JCL operands. If a user does not specify a retention period with RACF commands or JCL, RACF selects a retention period through profile modeling, an installation exit, or a system default set with the RETPD operand on the SETROPTS command.

If you have the SPECIAL attribute, you can establish a system default number of days with the RETPD operand. With this operand, you can specify a one to five digit number in the range of 0 through 65533. To set a default retention period for a data set that never expires, specify 99999. The following example shows how to specify a RACF security retention period of 365 days:

```
SETROPTS RETPD(365)
```

RACF uses the default security retention period for a tape data set in the following situations:

- When a user defines a data set (using ADDSD) without specifying a retention period
- When a user defines a data set (using ADDSD) or changes a data set profile (using ALTDSD) and specifies RETPD(0)
- When a user specifies RETPD=0 on the JCL statement
- When a user specifies EXPDT=today's date on the JCL statement
- When a user omits the RETPD and EXPDT parameters on the JCL statement

For example, if a user specifies RETPD=0 on the JCL statement and your installation has established a default retention period of 365 using SETROPTS RETPD, RACF uses 365 as the retention period for the user's data set.

The default security retention period when RACF is installed is RETPD(0), to indicate no retention period.

Notes:

1. The RACF security retention period is independent of the data set retention period specified by the EXPDT/RETPD JCL operands. However, the two retention periods are the same initially if the data set has a discrete profile. You can modify the security retention period by using the ALTDSD command, but you cannot change the data set retention period in the tape label of tape data sets.

2. The security retention period tape data sets has meaning only when both the TAPEVOL class and TAPEDSN are active.

Erasing Scratched or Released DASD Data (ERASE Option)

If you have the SPECIAL attribute, you can activate erase-on-scratch processing with the ERASE operand on the SETROPTS command. If erase-on-scratch is active and a DASD data set profile has the erase indicator set, ERASE specifies that data management is to erase the contents of any scratched or released data set extents that are part of a DASD data set protected by that profile. When RACF runs on a system that includes data management support for erase-on-scratch, the contents of a scratched and erased data set cannot be read.

The ERASE operand has several suboperands that allow an installation to override user specifications.

- ALL specifies that *all* data sets (including temporary data sets) are always erased, regardless of the erase indicator in the data set profile. When this option is selected, installation exit routines *cannot* prevent any data set from being erased by overriding this option.
- SECLEVEL allows you to specify a security level at which all data sets at this security level or higher are always erased, regardless of the erase indicator in the profile.
- NOSECLEVEL specifies that RACF is not to use the security level in the data set profile when it decides whether data management is to erase a scratched data set.

The following example shows how to activate erase-on-scratch processing for all data sets with a security level of CONFIDENTIAL or higher.

```
SETROPTS (ERASE) SECLEVEL(CONFIDENTIAL)
```

If you specify the ERASE operand without the ALL suboperand, erase-on-scratch processing applies only to DASD data sets that do not have system-generated temporary names and do not have names that begin with **SYSUT. You can extend erase-on-scratch to include temporary data sets with system-generated names by using the naming conventions table to modify system-generated names to look like permanent names. In this case, you need not specify ALL.

If you have the SPECIAL attribute, you can also deactivate erase-on-scratch processing by using the NOERASE operand on the SETROPTS command.

NOERASE is in effect when a RACF database is first initialized using IRRMIN00.

Establishing National Language Defaults (LANGUAGE Option)

If you have the SPECIAL attribute, you can specify the installation defaults for national languages on your system. You can specify a primary language and a secondary language. Applications that use the RACROUTE REQUEST=EXTRACT macro to determine a user's primary and secondary languages can use the installation defaults set by SETROPTS if the user does not have his or her own language preferences.

To specify the installation default languages, enter:

```
SETROPTS LANGUAGE(PRIMARY(language1) SECONDARY(language2))
```

You must specify a 3-character language code unless RACF is running with the MVS message service active.

LANGUAGE(PRIMARY(ENU) (SECONDARY(ENU)), which means American English, is in effect when a RACF database is first initialized using IRRMIN00.

SETROPTS Options to Activate In-Storage Profile Processing

RACF provides processing to activate in-storage profiles. It can help the administrator maximize performance of the RACF database. The SETROPTS operands are:

- GENLIST
- RACLIST

They allow your installation to reduce the amount of storage that is used by general resource profiles and the amount of processing that is associated with retrieving profiles from the RACF database, respectively.

Notes:

1. RACF does not allow you to specify SETROPTS GENLIST and SETROPTS RACLIST for the same general resource class at the same time.
2. Whenever you re-IPL the system, RACF automatically reactivates SETROPTS GENLIST and RACLIST processing for the classes for which this was previously requested.

SETROPTS GENLIST Processing

If you have the SPECIAL attribute, you can activate SETROPTS GENLIST processing. Activate this function for general resource classes that contain a small number of frequently referenced generic profiles. When you activate SETROPTS GENLIST processing, you enable the sharing of in-storage generic profiles for the classes you specify. For a list of the classes eligible for GENLIST processing, see the description of the class descriptor table (CDT) in *z/OS SecureWay Security Server RACF Macros and Interfaces*.

To activate this function, issue the SETROPTS GENLIST(*classname*), where *classname* is one of the following:

- A member class specified in the class descriptor table (CDT)
- Grouping class RACFVARS or NODES

RACF processes *classname* and all classes that share the same POSIT value on their class descriptor table (CDT) entries. The following example shows how to activate SETROPTS GENLIST processing for the TERMINAL class.

```
SETROPTS GENLIST(TERMINAL)
```

After you activate SETROPTS GENLIST processing for a general resource class, RACF copies a generic profile in that class from the RACF database into common storage the first time an authorized user requests access to a resource protected by it. The profile is retained in common storage and is available to all authorized users, thereby saving real storage because the need to retain multiple copies of the same profile (one copy for each requesting user) in common storage is eliminated. Also, because RACF does not have to retrieve the profile each time a user requires access to it, this function saves processing overhead.

Note that a general resource class must be active before you can activate SETROPTS GENLIST processing for that class. If the class is not active, issue the SETROPTS command with both the GENLIST and CLASSACT operands and specify the desired class. The following example shows how to activate the TERMINAL class and SETROPTS GENLIST processing for that class on the same command.

```
SETROPTS CLASSACT(TERMINAL) GENLIST(TERMINAL)
```

For more information on activating protection for specific general resource classes, check the index of this book for the class name.

Deactivating SETROPTS GENLIST Processing

If you have the SPECIAL attribute, you can deactivate SETROPTS GENLIST processing for general resource classes. To deactivate this function, issue the SETROPTS command with the NOGENLIST operand and the selected general resource classes.

NOGENLIST is the default and is in effect for all eligible classes that are defined in the class descriptor table (CDT) when a RACF database is first initialized using IRRMIN00.

See *z/OS SecureWay Security Server RACF System Programmer's Guide* for more information about SETROPTS GENLIST processing.

SETROPTS GENLIST Processing on Shared Systems

If your installation has two or more systems sharing a RACF database, you need to issue the SETROPTS GENLIST command on only one of those systems. SETROPTS GENLIST processing is automatically propagated to all systems sharing the database.

Refreshing Profiles for SETROPTS GENLIST Processing

If your installation has activated SETROPTS GENLIST processing for a particular resource class, you must refresh in-storage profiles for this processing when you make changes to one of these profiles in the database. Refreshing profiles for SETROPTS GENLIST processing ensures that the most current copy of a profile resides in common storage and is available for RACF authorization checking. To refresh profiles for this processing, issue the SETROPTS command with the GENERIC and REFRESH operands and specify the appropriate resource classes. For more information, see "Refreshing In-Storage Generic Profile Lists (GENERIC REFRESH Option)" on page 134.

For information about SETROPTS REFRESH processing on shared systems, see "Refreshing Shared Systems (REFRESH Option)" on page 135.

SETROPTS RACLIST Processing

If you have the SPECIAL attribute, you can activate SETROPTS RACLIST processing. When you activate SETROPTS RACLIST processing, you enable the sharing of both in-storage discrete and in-storage generic profiles for the classes you specify. For a list of the classes eligible for RACLIST processing, see the description of the class descriptor table (CDT) in *z/OS SecureWay Security Server RACF Macros and Interfaces*.

To activate this function, issue SETROPTS RACLIST(*classname*), where *classname* is one of the following:

- A member class for which RACLIST=ALLOWED is specified in the class descriptor table (CDT)
- Grouping class RACFVARS or NODES

RACF will RACLIST *classname* and all classes that share the same POSIT value on their class descriptor table (CDT) entries. The following example shows how to activate SETROPTS RACLIST processing for the TERMINAL class.

```
SETROPTS RACLIST(TERMINAL)
```

Notes:

1. If the system is enabled for sysplex communication and a command is successful on the system on which it was issued, RACF propagates the command to the other members of the data sharing group.
2. If the command fails on any of the peer systems and the system is in data sharing mode, RACF stops processing the command and backs it out of all the member systems, including the system on which it was issued.
3. In non-data sharing mode, the command can fail on a peer system without backing out of the other systems.
4. If the system is not enabled for sysplex communication, the command does not take effect on the other systems sharing the database until you issue it on those systems or the systems are IPLed.

When you activate SETROPTS RACLIST processing for a general resource class, RACF loads both discrete and generic profiles for the class into a data space. These profiles are available to all authorized users, thereby eliminating the need for RACF to retrieve a profile each time a user requests access to a resource protected by it. As a result, when you activate this function, you reduce processing overhead.

If the RACGLIST class is active and has a profile with the same name as the RACLISTed class, RACF saves the results on the database as *classname_nnnnn* profiles in the RACGLIST class, in addition to loading them into a data space. For example, RACF would save the RACLISTed data for the TERMINAL class as `TERMINAL_00001`, `TERMINAL_00002`, and so forth. For more information on RACGLIST, see “The RACGLIST Class” on page 275.

If RACROUTE REQUEST=LIST,GLOBAL=YES was previously issued for the class, issuing SETROPTS RACLIST deletes the data space created by the RACROUTE request and replaces it with a new one. The SETROPTS RACLIST overrides the GLOBAL=YES RACLIST. Output from a SETR LIST command displays the class in the `SETR RACLIST CLASSES =` line rather than in the `GLOBAL=YES RACLIST ONLY =` line. For more information, see “Using RACROUTE REQUEST=LIST,GLOBAL=YES Support” on page 274.

Note that a general resource class must be active before you can activate SETROPTS RACLIST processing for that class. If the class is not active, issue the SETROPTS command with both the RACLIST and CLASSACT operands and specify the desired class. The following example shows how to activate the TERMINAL class and SETROPTS RACLIST processing for that class on the same command.

```
SETROPTS CLASSACT(TERMINAL) RACLIST(TERMINAL)
```

For more information on activating protection for specific general resource classes, check the index of this book for the class name.

There are special considerations regarding profile size for resources in classes that are processed using SETROPTS RACLIST or RACROUTE REQUEST=LIST. See “Limiting the Size of Your Access Lists” on page 205.

Deactivating SETROPTS RACLIST Processing

If you have the SPECIAL attribute, you can deactivate SETROPTS RACLIST processing for general resource classes. To deactivate this option, issue SETROPTS NORACLIST(*classname*). RACF processes *classname* and all classes that share the same POSIT value on their class descriptor table (CDT) entries.

You can also use SETROPTS NORACLIST to delete a data space created by a RACROUTE REQUEST=LIST,GLOBAL=YES command. Therefore, the command must operate on classes even though RACLIST=DISALLOWED is specified in the class descriptor table (CDT). If RACGLIST is active and RACGLIST *classname_nnnnn* profiles exist, RACF deletes them and keeps the base RACGLIST profile name. For more information, see “The RACGLIST Class” on page 275 and “Using RACROUTE REQUEST=LIST,GLOBAL=YES Support” on page 274.

Notes:

1. You should issue a SETROPTS NORACLIST command for classes RACLISTed by RACROUTE REQUEST=LIST,GLOBAL=YES *only* after all classes that issued RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES have given up their access to the RACLIST data space by issuing RACROUTE REQUEST=LIST,ENVIR=DELETE.
2. If the system is enabled for sysplex communication and a command is successful on the system on which it was issued, RACF propagates the command to the other members of the data sharing group.
3. If the command fails on any of the peer systems RACF does *not* back it out of the member systems.
4. If the system is not enabled for sysplex communication, the command does not take effect on the other systems sharing the database until you issue it on those systems or the systems are IPLed.

NORACLIST is the default and is in effect for all eligible classes that are defined in the class descriptor table (CDT) when a RACF database is first initialized using IRRMIN00.

See *z/OS SecureWay Security Server RACF System Programmer’s Guide* for more information about SETROPTS RACLIST processing.

SETROPTS RACLIST Processing on Shared Systems

If two or more systems that are not enabled for sysplex communication are sharing a RACF database, SETROPTS RACLIST processing applies only to the system on which you issue the SETROPTS command. With this type of sharing, you must issue the SETROPTS command on all of the systems in order to perform RACLIST processing on all of the systems. If you do not issue the SETROPTS RACLIST command on one of the shared systems, RACLIST is performed for that system when the system re-IPLs.

On systems that are enabled for sysplex communication, issue the SETROPTS RACLIST command only once. If the command is successful, it is propagated to the other members of the data sharing group.

Refreshing Profiles for SETROPTS RACLIST Processing

Any changes made to discrete or generic profiles activated for SETROPTS RACLIST processing become effective only when you issue the SETROPTS command with both the RACLIST and REFRESH operands. You can refresh these profiles if you have one of the following:

- The SPECIAL attribute
- CLAUTH authority to the general resource class you specify on the RACLIST operand

To refresh the profiles, issue SETROPTS RACLIST(*classname*) REFRESH. RACF refreshes *classname* and all classes that share the same POSIT value on their class descriptor table (CDT) entries.

You can also use SETROPTS RACLIST REFRESH to refresh a class RACLISTed by a RACROUTE REQUEST=LIST GLOBAL=YES command. RACF deletes the old data space and loads the discrete and generic profiles for the class into a new data space.

SETROPTS RACLIST REFRESH has no effect on what line SETROPTS LIST displays a RACLISTed class. If the class were RACLISTed solely by RACROUTE REQUEST=LIST, ENVIR=CREATE, GLOBAL=YES the class will be listed in the `GLOBAL=YES RACLIST ONLY =` line. Regardless of whether the class was RACLISTed by that means, if it was RACLISTed by SETR RACLIST *classname*, the class will be listed only in the `SETR RACLIST CLASSES =` line.

If the RACGLIST class is active and contains a profile named *classname*, RACF rebuilds or creates the RACGLIST *classname_nnnnn* profiles to hold the new contents of the new data space. For more information, see “The RACGLIST Class” on page 275 and “Using RACROUTE REQUEST=LIST,GLOBAL=YES Support” on page 274.

Note that you must issue this command each time you want RACF to perform the refresh process. The following example shows how to activate refreshing of SETROPTS RACLIST processing for the DASDVOL and TERMINAL classes.

```
SETROPTS RACLIST(DASDVOL TERMINAL) REFRESH
```

For information about SETROPTS REFRESH processing on shared systems, see “Refreshing Shared Systems (REFRESH Option)” on page 135.

SETROPTS REFRESH Option for Special Cases

RACF provides the SETROPTS REFRESH operand to allow the administrator to refresh profiles in any situation.The options described in this section are:

- GENERIC REFRESH
- GLOBAL REFRESH
- REFRESH for shared systems

Refreshing In-Storage Generic Profile Lists (GENERIC REFRESH Option)

If you have the SPECIAL, AUDITOR, or OPERATIONS attribute, or if you have CLAUTH authority for the classes specified, you can initiate the refreshing of in-storage generic profile lists by specifying the GENERIC and REFRESH operands on the SETROPTS command.

When you specify GENERIC and REFRESH, you also specify one or more classes for which you want RACF to refresh in-storage generic profile lists. This causes all of the in-storage generic profiles in the specified general resource class (except those in the global access checking table) to be replaced with new copies from the RACF database. Note that you must issue this command each time you want RACF to perform the refresh process.

To refresh the profiles, issue the SETROPTS GENERIC(*classname*) REFRESH command. RACF processes *classname* and all classes that share the same POSIT value on their class descriptor table (CDT) entries.

The following example shows how to refresh in-storage generic profiles for the DATASET and TERMINAL classes.

```
SETROPTS GENERIC(DATASET TERMINAL) REFRESH
```

If you use SETROPTS GENLIST to activate shared in-storage generic profiles for a general resource class, RACF refreshes the profiles as well as the profile lists for that class when you specify the class with GENERIC and REFRESH. For more information, see “SETROPTS Options to Activate In-Storage Profile Processing” on page 130.

If you specify GENERIC(*), RACF refreshes profile lists for the DATASET class and all active classes in the class descriptor table (CDT) except group resource classes.

If you specify NOGENERIC on the SETROPTS command, RACF stops using in-storage generic profile lists but does not immediately delete them. RACF deletes the profile lists at the end of the job or TSO session, or when you again specify GENERIC. When you specify GENERIC, RACF rebuilds the profile lists.

Note: You must have the SPECIAL attribute to issue the SETROPTS GENERIC command by itself. However, to issue SETROPTS GENERIC (*classname*) REFRESH, you do not need the SPECIAL attribute. However, you must have the group-SPECIAL, group-AUDITOR, group-OPERATIONS, AUDITOR, or OPERATIONS attribute, or you must have CLAUTH authority for the classes specified.

For information about SETROPTS REFRESH processing on shared systems, see “Refreshing Shared Systems (REFRESH Option)”.

Refreshing Global Access Checking Lists (GLOBAL REFRESH Option)

If you have the SPECIAL attribute, you can initiate the refreshing of global access checking lists by specifying the GLOBAL and REFRESH operands on the SETROPTS command. When you specify GLOBAL and REFRESH, also specify the class for which you want RACF to refresh global access checking lists. Note that you must issue this command each time you want RACF to perform the refresh process.

To refresh the profiles, issue the SETROPTS GLOBAL(*classname*) REFRESH command. RACF processes *classname* and all classes that share the same POSIT value on their class descriptor table (CDT) entries.

The following example specifies that you want RACF to refresh global access checking lists for the DATASET and TERMINAL classes.

```
SETROPTS GLOBAL(DATASET TERMINAL) REFRESH
```

If you specify GLOBAL(*), RACF refreshes access checking lists for the DATASET class and all active classes in the class descriptor table (CDT).

If you specify NOGLOBAL, you disable global access checking for the class that you specify.

For information about SETROPTS REFRESH processing on shared systems, see “Refreshing Shared Systems (REFRESH Option)”.

Refreshing Shared Systems (REFRESH Option)

If two or more systems that are *not* enabled for sysplex communication are sharing a RACF database, SETROPTS REFRESH processing applies only to the system on which you issue the SETROPTS command. With this type of sharing, you must issue the SETROPTS command on all of the systems to perform REFRESH

processing on all of the systems. If you do not issue the SETROPTS REFRESH command on one of the shared systems, REFRESH is performed for that system when the system re-IPLs.

On systems that are enabled for sysplex communication, issue the REFRESH command only once. If the command is successful, it is propagated to the other members of the data sharing group.

If the command fails on any of the peer systems and the system is in data sharing mode, RACF stops processing the command and backs it out of all the member systems, including the system on which it was issued.

In non-data sharing mode, the command can fail on a peer system without backing out of the other systems.

SETROPTS Options for Special Purposes

Some options are useful for special purposes. You can specify them at any time, but you might not want to specify them when RACF is first installed. These options include:

- TERMINAL
- CLASSACT for the SECDATA and SECLABEL classes
- SESSIONINTERVAL
- MLQUIET
- WHEN(PROGRAM)

Protecting Undefined Terminals (TERMINAL Option)

If you have the SPECIAL attribute, you can specify the universal access authority (UACC) that RACF uses when users attempt to log on to TSO from terminals that are not defined to RACF. You can specify this option with the TERMINAL operand of the SETROPTS command, as shown in the following example:

```
SETROPTS TERMINAL(READ|NONE)
```

Attention

Before you specify NONE, be sure that you define your terminals to RACF and give the appropriate users and groups proper authorization to use them. Otherwise, users cannot access the terminals.

When you specify READ or NONE, you establish a default UACC for all users to undefined terminals on your system. If you specify READ, all users can access all terminals on your system (if allowed to by the security classifications of the terminals). If you specify NONE, only users and groups that you authorize to use a terminal through its access list can use it. If you do not specify either READ or NONE, the default value is READ. For more detailed information, see “Protecting Terminals” on page 235.

For undefined terminals, READ access authority is in effect when a RACF database is first initialized using IRRMIN00.

Activating the Security Classification of Users and Data

If you have the SPECIAL attribute, you can activate security classification of users and data by specifying CLASSACT(SECDATA) or CLASSACT(SECLABEL) on the SETROPTS command, as shown in the following examples:

```
SETROPTS CLASSACT(SECDATA)
SETROPTS CLASSACT(SECLABEL)
```

Security classification of users and data allows an installation to impose additional access controls on resources by defining security levels, security categories, and security labels for both users and resources.

Note: You can create profiles in the SECDATA and SECLABEL classes, and specify security classifications in resource and user profiles, without actually activating the SECDATA and SECLABEL classes. When authorization checking occurs, the security classifications are ignored by RACF. This allows you to “label” the profiles without enforcing the security classifications.

If you have the SPECIAL attribute, you can also deactivate security classification of users and data by specifying NOCLASSACT(SECDATA) or NOCLASSACT(SECLABEL), as appropriate, on the SETROPTS command.

NOCLASSACT(SECDATA) and NOCLASSACT(SECLABEL) are in effect when a RACF database is first initialized using IRRMIN00.

Establishing the Maximum VTAM Session Interval (SESSIONINTERVAL Option)

If you have the SPECIAL attribute, you can set the maximum number of days that any session segment password in an APPCLU profile can go without being changed. If any APPCLU profile has a higher interval limit, the SETROPTS value is used instead.

To set the limit, enter:

```
SETROPTS SESSIONINTERVAL(n)
```

where *n* is the maximum number of days that can be specified and must be between 1 and 32767.

To remove the system-wide maximum (allowing any value in the profiles to take effect), enter:

```
SETROPTS NOSESSIONINTERVAL
```

Quiescing RACF Activity (MLQUIET Option)

If you have the SPECIAL attribute, you can prevent users other than SPECIAL users, console operators, and started procedures from logging on, starting new jobs, or accessing resources. This prevents them from using the RACROUTE AUTH, DEFINE, and VERIFY requests.

To do this, enter:

```
SETROPTS MLQUIET
```

To cancel this option, specify NOMLQUIET on the SETROPTS command.

Attention

Do not specify SETROPTS MLQUIET if any system using the RACF database is not at the necessary software level for B1 support. For a list of the products that must be installed, see “B1 Configuration Requirements” on page 12.

Use of the MLQUIET option can prevent a successful IPL of these systems. In addition, if no systems using the RACF database are capable of B1 support, you may have to IPL with RACF inactive and update the database with the block update command (BLKUPD) in order to turn off the MLQUIET option.

Activating Program Control (WHEN(PROGRAM) Option)

If you have the SPECIAL attribute, you can activate program control by using the WHEN(PROGRAM) operand of the SETROPTS command. When program control is active, RACF provides both access control to load modules and program access to data sets. The following example shows how to specify this option:

```
SETROPTS WHEN(PROGRAM)
```

Access control to load modules allows only authorized users to load and execute specified load modules (programs). RACF uses profiles in the PROGRAM general resource class to control access to programs.

Program access to data sets allows an authorized user or group of users to access specified data sets in conjunction with the user’s authority to execute a certain program. That is, some users can access specified data sets at a specified access level only while executing a certain program.

If you have the SPECIAL attribute, you can also deactivate program control by using the NOWHEN(PROGRAM) operand on the SETROPTS command.

NOWHEN(PROGRAM) is in effect when a RACF database is first initialized using IRRMIN00.

Notes:

1. If the system is enabled for sysplex communication and a command is successful on the system on which it was issued, RACF propagates the command to the other members of the data sharing group.
2. If the command fails on any of the peer systems and the system is in data sharing mode, RACF stops processing the command and backs it out of all the member systems, including the system on which it was issued.
3. If the system is not enabled for sysplex communication, the command does not take effect on the other systems sharing the database until you issue it on those systems or the systems are IPLed.
4. In non-data sharing mode, the command can fail on a peer system without backing out of the other systems.

SETROPTS Options Related to Security Labels

Several options are related to security labels. These options require the SECLABEL class to be active. They include:

- SECLABELCONTROL
- MLS
- COMPATMODE

- MLACTIVE
- MLSTABLE

Restricting Changes to Security Labels (SECLABELCONTROL Option)

If you have the SPECIAL attribute, you can prevent users who do not have the SPECIAL attribute from doing either of the following:

- Specifying or changing a security label in a resource profile
- Changing the profiles named SECLEVEL or CATEGORY, or changing any profile in the SECLABEL class, such that the definition of a security label changes

To place this control into effect, enter:

```
SETROPTS SECLABELCONTROL
```

When the SECLABELCONTROL option is in effect, only certain users can specify the SECLABEL operand on RACF commands:

- Users with the SPECIAL attribute can specify the SECLABEL operand on any RACF command.
- Users with the group-SPECIAL attribute can specify the SECLABEL operand only on the ADDUSER and ALTUSER commands when they add a user to a group within their scope of control. Also, group-SPECIAL users must be permitted to the SECLABEL profiles with at least READ access authority.
- Users without the SPECIAL attribute cannot specify the SECLABEL operand.

To cancel this option, specify NOSECLABELCONTROL on the SETROPTS command.

Preventing the Copying of Data to a Lower Security Label (MLS Option)

If you have the SPECIAL attribute, and if the SECLABEL class is active, you can prevent users from copying data from a resource with one security label to a resource with a lower security label. Users can still copy data from a lower security label to the security label the user is currently logged on with.

To do this, enter:

```
SETROPTS MLS(FAILURES)
```

You can also specify MLS(WARNING), which allows the user request, but sends a warning message to the user and the security administrator.

To cancel the MLS option, specify NOMLS on the SETROPTS command.

Note: Do not specify SETROPTS MLS if any system using the RACF database is not at the necessary software level for B1 support. Use of the MLS option should not cause problems on these systems, but it does not provide full protection on these systems. For a list of the products that must be installed, see “B1 Configuration Requirements” on page 12.

Activating Compatibility Mode for Security Labels (COMPATMODE Option)

If you are using SECLABELs on your system, and you observe SECLABEL failures for some calls at the designated security console or in audit records, the reason may be that the caller was not using or was unable the RACF protocols.

To investigate the source of the failure, obtain a copy of the RACF Report audit record. Examine the EVENT and QUALIFIER fields for the call to see if the failure occurred because of insufficient security-label authority. Next examine the “token status =” field to the right of the QUALIFIER field. If the field is identified as being created by a call that was either not using or unable to use the current protocols, RACF failed the SECURITY label authorization check.

If this was the case, and you want to have SECLABEL authorization checks succeed for those callers who are not using current protocols, you may be able to use the COMPATMODE option on the SETROPTS command to do so. Specifying COMPATMODE allows the caller to access the resources it needs, providing the user is in the access list of a SECLABEL (it need not be one that protects the resource) that is higher than or equal to the SECLABEL that protects the resource.

To establish COMPATMODE, enter:

```
SETROPTS COMPATMODE
```

NOCOMPATMODE is in effect when a RACF database is first initialized using IRRMIN00.

Enforcing Multilevel Security (MLACTIVE Option)

If you have the SPECIAL attribute, and if the SECLABEL class is active, you can fully enforce multilevel security. To check your system configuration, see “B1 Configuration Requirements” on page 12. Multilevel security requires the following:

- All work entering the system must be run by a RACF-defined user.
- A security label must be assigned to all work entering the system, including batch jobs and users logging on to TSO and MVS consoles, and to any application that supports security labels when users log on.
- A security label must be assigned to all profiles in the following classes:
  - APPCPORT
  - APPCSERV
  - APPCTP
  - DEVICES
  - DATASET
  - USER
  - TAPEVOL
  - TERMINAL
  - WRITER

  To do this, enter:

  ```
  SETROPTS MLACTIVE
  ```

  You can also specify MLACTIVE(WARNING), which allows the users to log on or submit jobs. MLACTIVE(WARNING) sends a warning message to the user and to the security administrator when the user attempts to:

  - Enter the system without a security label
  - Access a resource in one of the previously mentioned classes but the resource has not been assigned a security label

  To cancel the MLACTIVE option, specify NOMLACTIVE on the SETROPTS command.

Note: Do not specify SETROPTS MLACTIVE if any system using the RACF database is not at the necessary software level for B1 support. For a list of the products that must be installed, see “B1 Configuration Requirements” on page 12.

Use of the MLACTIVE option should not cause problems on these systems if you assign a default security label to all users and to all resources in the classes listed above. However, because users on these systems cannot supply a security label when they log on or run a batch job, the usefulness of requiring that everything be labelled is reduced. Also, some error recovery scenarios may require the security administrator to log on specifically at the SYSHIGH level, and this is not possible unless at least one system sharing the database has the B1-support software.

Attention

Do not issue the SETROPTS MLACTIVE(FAILURES) command unless you have assigned appropriate security labels to users and to the resources that they must access. To recover from such a situation, logon as a user with the SPECIAL attribute, specifying SYSHIGH as the current security label. Then either assign security labels, or issue SETROPTS NOMLACTIVE.

Preventing Changes to Security Labels (MLSTABLE Option)

If you have the SPECIAL attribute, you can prevent users from changing the classification of data while the data is in use. Specifically, you can do all of the following:

- Prevent any user from changing the security label of a RACF profile
- Prevent any user from changing a SECLABEL profile such that the definition of the security label changes. (Users cannot change the security level or security categories associated with the security label.) Other changes to the SECLABEL profile, such as changes to the access list, are still allowed.

To do this, enter:

```
SETROPTS MLSTABLE
```

To cancel this option, specify NOMLSTABLE on the SETROPTS command.

Note: If you must change security labels while the system is in multilevel stable state, you can issue SETROPTS MLQUIET before making the changes. See “Quiescing RACF Activity (MLQUIET Option)” on page 137,

SETROPTS Options for Automatic Control of Access List Authority

Use of the SETROPTS options ADDCREATOR and NOADDCREATOR allows you to specify whether the user ID of the person defining a resource profile is automatically placed on the access list for that resource with ALTER authority. The options are:

- ADDCREATOR
- NOADDCREATOR

Automatic Addition of Creator's User ID to Access List

The ADDCREATOR option indicates that the profile creator's user ID is placed on the profile access list with ALTER authority when any new DATASET or general resource profiles is defined using ADDSD, RDEFINE, or RACROUTE REQUEST=DEFINE.

This option is the default unless IRRMIN00 is run with PARM=NEW.

Automatic Omission of Creator's User ID from Access List

The NOADDCREATOR option indicates that the profile creator's user ID is not placed on the profile access list with ALTER authority under the following conditions:

- When any new DATASET or general resource profiles is defined using ADDSD, RDEFINE, or creating a generic profile through RACROUTE REQUEST=DEFINE
- When a discrete profile (other than the DATASET and TAPEVOL classes) is created through RACROUTE REQUEST=DEFINE

Even if the NOADDCREATOR option is used, in the DATASET and TAPEVOL classes created through RACROUTE REQUEST=DEFINE, the user ID of any profile creator is placed on the new profile's access list with ALTER authority.

This will occur when a user creates a permanent data set, if the user has ADSP and ADSP is active on the system, or when a user specifies PROTECT or SECMODEL on the JCL DD statement or TSO allocate command for a new permanent data set.

If IRRMIN00 is run with PARM=NEW, this option is the default.

Specifying the Encryption Method for User Passwords

By default, RACF uses the data encryption standard (DES) algorithm to encrypt passwords and operator identification card (OIDCARD) data.

If you want to use the ICHDEX01 exit routine to store the passwords and OIDCARD data in a masked form, use one of the following methods to override the DES algorithm:

- Use MLPA to cause RACF to find the exit. This is the recommended method because you only need to do it once.
- Create SMP.E USERMOD to claim ownership of ICHDEX01 and move it to LPALIB. This is not recommended because you need to perform this operation with each installation.

RACF performs two different encoding functions:

- Password/OIDCARD data encoding
- Password/OIDCARD data comparison

Encoding means that, given data in clear text and given an encryption key (which RACF constructs), the equivalent data is produced in encrypted form. RACF provides a "one-way" encoding. That is, data encrypted by RACF can only be decoded if the data is already known. For additional details, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Comparison means that, given a password (or OIDCARD data) as entered by a user (in clear text form) and given a password (or OIDCARD data) as stored in the RACF database in encoded form, an indication as to whether they are equal or not is returned.

RACF performs password comparison in the following way:

- RACF encrypts the user-entered data using the DES algorithm and compares it against the stored version. If they are equal, RACF returns to the caller with an “equal” indication.
- RACF encodes the user-entered data using the current masking algorithm and compares it against the stored version. If they are equal, RACF returns to the caller with an “equal” indication.

By encoding the user-entered data against both the DES algorithm and the masking algorithm, RACF allows the use of existing masked passwords and OIDCARD data until they can be replaced by the DES forms.

For compatibility with previous versions of RACF, a dummy ICHDEX01 exit routine is supplied with RACF. You should delete the dummy exit routine on all systems that share the RACF database after all of these systems have been converted to a version of RACF that supports the DES algorithm.

Using Started Procedures

A *procedure* consists of a set of job control language statements that are frequently used together to achieve a certain result. Procedures usually reside in the system procedure library, SYS1.PROCLIB, which is a partitioned data set. A *started procedure* is usually started by an operator, but can be associated with a functional subsystem. For example, DFSMS is treated as a started procedure even though it does not need to be specifically started with a START command.

Only RACF-defined users and groups can be specifically authorized to access RACF-protected resources. However, started procedures have system-generated JOB statements that do not contain the USER, GROUP, or PASSWORD parameter.

To enable started procedures to access the same RACF-protected resources that users and groups access, started procedures must have RACF user IDs and group names. By assigning them RACF identities, your installation can give started procedures specific authorization to access RACF-protected resources. For example, you can allow JES to access spool data sets. To associate the names of started procedures with specific RACF group names and user IDs, you and your RACF system programmer can do one of the following:

1. Set up the STARTED class (the recommended method).
2. Create a started procedures table (ICHRIN03).

Assigning RACF User IDs to Started Procedures

As with any other user ID and group name, the user ID and group name that you assign to a started procedure must be defined to RACF using the ADDUSER and ADDGROUP commands, and the user must be connected to the group. You may also need to use the PERMIT command to authorize the users or groups to get access to the required resources. See *z/OS SecureWay Security Server RACF Command Language Reference* for descriptions of these commands.

Protected User IDs

The user IDs that you assign to started procedures should have the PROTECTED attribute. Protected user IDs are user IDs that have both the NOPASSWORD and NOOIDCARD attributes. They are defined or modified using the ADDUSER and ALTUSER commands. See “Defining Protected User IDs” on page 86 for more information.

Protected user IDs can not be used to logon to the system, and are protected from being revoked through incorrect password attempts. The following example shows a protected user ID being defined for a CICS region, and an existing user ID used by JES being given the PROTECTED attribute:

```
ADDUSER CICS03 DFLTGRP(STCGROUP) OWNER(STCADMIN) NOPASSWORD
ALTUSER JES    DFLTGRP(STCGROUP) OWNER(STCADMIN) NOPASSWORD
```

If you do not specify NOPASSWORD for a user ID assigned to a started procedure, you should specify a password and change the password periodically. If you do not specify a password and do not specify NOPASSWORD, RACF uses the default group name as the password. Anyone who knows this user ID and password combination can gain access to any resource that the started procedure can access.

See “Using Protected User IDs for Batch Jobs” on page 444 for more information.

Note: If the associated user ID is revoked for any reason, the started procedure may have problems allocating new SMS-managed data sets, submitting batch jobs, and obtaining printed output.

Undefined User IDs

A started procedure runs as an undefined user if:

1. It is executed without associating its name with a RACF-defined user ID and group name.
2. The user or group is not defined.
3. The user is not connected to the group.

A started procedure running as an undefined user can access RACF-protected resources if the universal access authority for the resource is sufficient to allow the requested operation. However, if a started procedure requires access at a higher level than universal access, you *must* associate the started procedure with a RACF-defined user ID and group name.

Authorizing Access to Resources

A started procedure can gain access to RACF-protected resources in the following ways:

1. By the user ID or group name assigned as for any other user of the system (for example, universal access, entry and access list, and OPERATIONS).
2. By having the privileged attribute, which allows the started procedure to pass all authorization checking (unless the CSA or PRIVATE operand is specified on the RACROUTE request). No installation exits are called, no SMF records are generated, and no statistics are updated. (Note that bypassing authorization checking includes bypassing the checks for security classification of users and data.)
3. By having the trusted attribute, which means the same as privileged, except that you can request an audit using the SETROPTS LOGOPTIONS command.

Setting Up the STARTED Class

With the STARTED class, you don't need to change code or re-IPL the system in order to add or modify RACF identities for started procedures. You can modify the security definitions for started procedures *dynamically,* using the RDEFINE, RALTER, and RLIST commands. See *z/OS SecureWay Security Server RACF Command Language Reference* for more information on these commands. In effect, the STARTED class provides a *dynamic* started procedures table.

To set up the STARTED class, enter these commands (for example):

```
SETROPTS GENERIC(STARTED)

RDEFINE STARTED JES2.* STDATA(USER(JES2)    GROUP(STCGROUP) TRUSTED(YES))
RDEFINE STARTED **     STDATA(USER(=MEMBER) GROUP(STCGROUP) TRACE(YES))

SETROPTS CLASSACT(STARTED)
SETROPTS RACLIST(STARTED)

 or, if changes have been made to the STARTED class profiles:

SETROPTS RACLIST(STARTED) REFRESH
```

Defining Profile Data

Profiles in the STARTED class include the STDATA segment, which contains fields for user ID, group name, trusted flag, privileged flag, and trace flag:

- The user ID can be a RACF user ID or the character string `=MEMBER`, which indicates that the member name is to be used as the user ID.
- The group name can be a RACF group name or the character string `=MEMBER`, which indicates that the member name is to be used as the group name.
- If tracing is specified, RACF issues operator message IRR812I during RACROUTE REQUEST=VERIFY or VERIFYX to indicate which profile is used.

  This message can be used during diagnosis of security problems with started procedures, to determine which profile was used for a particular started procedure.

RACF performs partial diagnosis when creating the STDATA segment to help you define profiles that work correctly. For example, RACF verifies that a specified user ID is connected to the group name, if specified.

Attention

Be sure to specify a group name (not `=MEMBER`) as the GROUP value of the STDATA segment, if both of the following are true:

1. The profile name contains generic characters (*,%, or &).
2. The USER value of the STDATA segment is the character string `=MEMBER`.

If you do not specify a group name, a new started procedure or job could be assigned on execution to a user ID that matches an existing user ID on your system. Consider defining a special group (for example, STCGROUP) for started procedures and job user IDs, and using this group name as the GROUP value of the STDATA segment.

In addition, be careful which libraries your started procedures come from and do not let your users update them. Refer to the JES customization manuals for information on specifying procedure libraries.

Specifying STARTED Class Profile Names

With MVS/ESA 5.1 or later, you can start jobs as well as procedures using the START command. The START command specifies the member name to start and, optionally, the job name to use. See *z/OS MVS System Commands* for more information on the START command.

Resource names in the STARTED class are of the form *member.job*, where:

member is the 1–8 character name of a member of a partitioned data set that contains the source JCL for the task to be started. The member can be a job or a cataloged procedure.

job is the name identifying the task to be started. If the START command does not specify a job name, and the member does not contain a JOB statement to supply a job name, the system uses the member name as the job name when constructing the resource name for STARTED class processing.

For each sample START command issued, Table 9 shows the resource name that will be used for STARTED class processing, and a list of names for STARTED class profiles that could be defined for each STARTED class resource.

Table 9. Sample Profile Names for STARTED Class Resources

START Command	STARTED Class Resource Name	STARTED Class Profile Name
S CICS	CICS.CICS	CICS.CICS
		CICS.*
		CICS.**
S CICSP	CICS.CICSP	CICSP.CICSP
		CICSP.*
		CICSP.**
		CICS*.**
		CICS*
S CICST	CICS.CICST	CICST.CICST
		CICST.*
		CICST.**
		CICS*.**
		CICS*
S IMS,JOBNAME=IMSPROD	IMS.IMSPROD	IMS.IMSPROD
		IMS.IMSP*
		IMS.*
		IMS.**
S IMS,JOBNAME=IMSTEST	IMS.IMSTEST	IMS.IMSTEST
		IMS.IMST*
		IMS.*
		IMS.**

Using the Started Procedures Table (ICHRIN03)

Your RACF system programmer can use the started procedures table (ICHRIN03) to associate the names of started procedures with specific RACF user IDs and group names.

The started procedures table can also contain a generic entry that assigns a user ID or group name to any started task that does not have a matching entry in the table. In this case, the table can specify that the started task name should be used as the user ID or group name, or it can assign a specific user ID or group name to the started task. You should ensure that the generic entry, if used, assigns a generic user ID or group name (for example, STCGROUP).

To modify the security definitions for started procedures using the started procedures table, you need to:

1. Edit the started procedures table.
2. Assemble and link-edit the updated table.
3. Re-IPL the system.

See *z/OS SecureWay Security Server RACF System Programmer's Guide* for information on how to code the started procedures replaceable module, and for a complete description of the started procedures table (ICHRIN03).

Started Procedure Considerations

Here are some things to consider when you use started procedures:

1. Even if your installation uses the STARTED class, you must have a started procedures table (ICHRIN03). RACF cannot be initialized if ICHRIN03 is not present. A dummy ICHRIN03 is shipped with and installed by RACF. If you use the STARTED class, you should leave your existing ICHRIN03 in place, in case, for example, someone unintentionally deactivates the STARTED class. For more information, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.
2. For installations that have an existing started procedures table(ICHRIN03) and want to use the STARTED class, a sample REXX exec is provided in member ICHSPTCV in SYS1.SAMPLIB to process the output of ICHDSM00 and build RDEFINE commands to duplicate an existing started procedures table.
3. To make sure that critical system tasks (those marked TRUSTED or PRIVILEGED in ICHRIN03) start successfully, define specific STARTED profiles for them in the STARTED class.
4. The TRUSTED attribute can be assigned to key started procedures that have the capability of preventing a successful IPL and those that need to create or access a wide variety of unpredictably named data sets. Candidates for the TRUSTED attribute include: JES, LLA, CATALOG, DUMPSRV, IEEVMPCR, SMF, VLF, VTAM, APSWPROC, RACF (if RRSF is used), and both IXGLOGR and XCFAS (if sysplex communication is used).
5. When the STARTED class is active, RACF uses it before using the started procedures table (ICHRIN03). A generic profile such as ****** or ***.*** with a valid STDATA segment will override all the entries in ICHRIN03.
6. To make sure that RACF uses the STARTED class, you should verify that all START commands have a matching profile with an STDATA segment that assigns a user ID. To do this:
   a. Define an appropriate generic profile that matches all possible START commands (for example, ****** or ***.***).
   b. Specify `=MEMBER` or a user ID of limited privileges.

c. Specify a group name, if you have specified =MEMBER as the USER value.

This approach ensures that, for any START command, there is always a matching profile with an STDATA segment that assigns a user ID. In addition, using this approach avoids the following situations, which cause RACF to use ICHRIN03 to process the START command:

a. There is no matching profile.
b. There is a matching profile, but it does not have an STDATA segment.
c. There is a matching profile with an STDATA segment, but no user ID is specified.
d. There is a matching profile with an STDATA segment, no user ID is specified, but the assigned user ID matches an existing user ID on your system.

7. When RACROUTE REQUEST=VERIFY or VERIFYX is issued with a started procedure name, RACF checks to see if the STARTED class is active. If it is active, RACF uses the STARTED class to determine the user ID, group name, trusted flag, and privileged flag to use. If the STARTED class is not active, RACF uses the started procedures table (ICHRIN03). RACF also uses the started procedures table, and issues message IRR813I or IRR814I if the STARTED class is active but one of the following occurs:
   a. RACF cannot find a matching profile in the STARTED class.
   b. RACF finds a matching profile but the profile does not assign a user ID.

Chapter 6. Protecting Data Sets on DASD and Tape

This chapter contains in-depth information on protecting data sets on DASD and tape.

Protecting Data Sets

This section describes considerations related to using RACF to protect data sets on DASD and tape. Unless there is an explicit qualification, all of the information in this section applies to both DASD and tape.

To protect data sets, create data set profiles. These profiles can be either discrete or generic. See *z/OS SecureWay Security Server RACF General User's Guide* for specific procedures to use in creating data set profiles, including examples of how to use the ADDSD and PERMIT commands. See *z/OS SecureWay Security Server RACF Command Language Reference* for complete descriptions of the ADDSD and PERMIT commands.

Automatic direction of application updates for the DATASET class require special consideration. See "Controlling Automatic Direction of Passwords" on page 286 and "Considerations for the DATASET Class" on page 382.

Rules for Defining Data Set Profiles

When you define data set profiles to RACF, you can use either standard or nonstandard naming conventions. If you use nonstandard naming conventions, the data set naming convention table and the single-level data set names option are ways to help “fit” RACF standard naming conventions.

The descriptions of naming conventions are followed by rules for protecting and allocating user and group data sets.

Standard Data Set Naming Conventions

By default, RACF expects a data set name (and the data set profile name) to consist of at least two qualifiers. RACF also expects the high-level qualifier of the data set profile name to be either a RACF-defined user ID or a RACF-defined group name.

If you and your implementation team have chosen to define data set profiles under the standard RACF naming conventions, you can create a group for each high-level qualifier that is not a user ID, and permit users to protect any data set that has that high-level qualifier by giving them CREATE authority in that group.

RACF can help enforce standard naming conventions at your location in several ways. These ways require users to use your predefined naming convention so that their data sets are RACF-protected.

- RACF has a PROTECTALL option on the SETROPTS command that allows a user to create or access a data set only if the data set is RACF-protected, by either a discrete or generic profile. See “RACF-Protecting All Data Sets (PROTECTALL Option)” on page 123 for more information.
- If your installation does not use PROTECTALL, use a RACROUTE REQUEST=DEFINE exit routine to ensure that a predefined generic profile exists before allowing a user to create a data set.
- When your users have the ADSP attribute, they can create or protect only data sets whose names begin with their own user ID, or for which they have CREATE or higher authority in the RACF group corresponding to the high-level qualifier of the data set name.

Table-Driven Data Set Naming Conventions

You can use the naming convention table to set up and enforce a data set naming convention other than that used by RACF. The table can:

- Supply a qualifier to be used as the high-level qualifier for authorization checking
- Convert data set names to RACF naming convention form for RACF use
- Convert names in RACF form to the installation’s format for external display
- Enforce a naming convention by not allowing the definition of data sets that do not conform to an installation’s rules
- Reduce RACF overhead by determining whether a data set is a user or group data set

You can create a naming convention table (module ICHNCV00), which RACF uses to check and modify (internally to RACF) the data set name in all commands and macros that process data set names. You can use the table to selectively rearrange data set names to “fit” the RACF convention without actually changing those names.

Naming convention processing is done by RACF immediately before the preprocessing/naming convention installation exits are called. (The exits can still be used for additional processing.)

The RACF table-driven naming convention feature largely replaces the need for the ICHCNX00 exit routine. (The naming convention table is processed before each call to ICHCNX00.)

For more information on creating and using the RACF naming convention table, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Protecting Data Sets That Have Single-Qualifier Data Set Names

If some of the data sets in your installation have names that consist of a single qualifier, you can still RACF-protect those data sets. To get RACF protection for single-qualifier names, issue the SETROPTS command with the PREFIX operand. This command defines a high-level qualifier to be used as a prefix for single-qualifier names and activates the facility. Then, when RACF processes requests for the data set, RACF internally modifies single-qualifier names by adding the prefix, making the data set names acceptable to RACF routines.

In subsequent references to the profile, all RACF commands and the RACF report writer expect to see the prefix followed by a period and the single-level data set name. All SMF log records and all messages from RACF contain the RACF-modified version of the data set name.

Attention

If you do not issue the SETROPTS command with the PREFIX operand, a system ABEND occurs when a user attempts to create a data set with a single-qualifier name. This abend occurs only when creating a discrete profile as part of data set allocation.

Note: The real data set names option (specified by the REALDSN operand on the SETROPTS command) applies only to name conversions made by the naming conventions table or installation exit routines. This option has no effect on single-qualifier data set names (unless they have been modified by the naming conventions table or an exit routine), whose "real data set names" continue to be the prefixed ones.

For more information on specifying the prefix, see "Protecting Data Sets with Single-Qualifier Names (PREFIX Option)" on page 126.

Protecting User Data Sets

A *user data set* is a data set whose high-level qualifier is a RACF user ID. The following rules apply to user data sets:

- In general, all RACF-defined users can protect their own data sets. However, some SETROPTS options can restrict the ability of users to define and change profiles. See "Restricting Changes to Security Labels (SECLABELCONTROL Option)" on page 139.
- A user can RACF-protect a data set for another user under any of the following conditions:
  - The user who is protecting the data set has the SPECIAL attribute. A discrete or generic profile can be created.

  - The user who is protecting the data set has the group-SPECIAL attribute, and the high-level-qualifier of the data set name is a user within the group-SPECIAL user's scope of authority. A discrete or generic profile can be created.
  - The user who is protecting a data set has the OPERATIONS attribute (or the group-OPERATIONS attribute if the data set is within his scope of authority) **and** is simultaneously creating the data set.

    In this case, the user can create a discrete profile:

    - Through ADSP
    - By specifying the PROTECT operand on the TSO ALLOCATE command that creates the data set
    - By specifying the PROTECT=YES OR SECMODEL=*profile-name* operands on the JCL DD statement that creates the data set
- The REQUEST=DEFINE preprocessing exit routine allows RACF protection.

Protecting Group Data Sets

A *group data set* is a data set whose high-level qualifier is a RACF group name. A RACF-defined user can RACF-protect a group data set under any of the following conditions:

- The user has JOIN, CONNECT, or CREATE authority in the group.
- The user has the SPECIAL attribute (or the group-SPECIAL attribute for that group) and the request is made using the ADDSD command.
- The user has the OPERATIONS attribute and is not connected to the group.
- The REQUEST=DEFINE preprocessing exit routine is used to override normal RACF authorization requirements.

Controlling the Creation of New Data Sets

Using data set profiles, you can control whether users can create (allocate) new data sets.

For cataloged data sets, creating, deleting, or renaming the data set involves access not only to the data set profile protecting the data set, but also to the catalog in which the data set is cataloged. In general, users need the following:

- To add entries to the catalog, users need authority to create the data set as specified below and UPDATE authority to the catalog.
- To delete entries from the catalog, users need ALTER authority to the protecting profile or to the catalog.

For more information, see "Protecting Catalogs" on page 172 and *z/OS DFSMS: Managing Catalogs*.

The following cases describe how RACF can be used to control the creation of new user and group data sets.

A user can create a new *user* data set in the following situations:

- The data set is protected by an existing generic profile and the user does not have ADSP.

  The creation is allowed if (1) the user has ALTER authority to the data set through the generic profile or global access checking, or (2) the data set is the user's own data set. RACF does not create a profile.
- The data set name is not covered by an existing generic profile and the user does not have ADSP.

If PROTECTALL is not in effect, the creation is allowed, but RACF does not create a profile. See Note 2.

- The user has ADSP and the data set is the user's own data set.

  The creation is allowed and RACF creates a discrete profile for the data set.
- The REQUEST=DEFINE preprocessing exit routine allows RACF protection.
- The user has the OPERATIONS attribute. If the user has the group-OPERATIONS attribute (that is, the user is connected to a group with the OPERATIONS attribute), the high-level qualifier of the new data set must be the ID of a user who is within the scope of that group.

A user can create a new *group* data set in the following situations:

- The data set name is protected by an existing generic profile and the user does not have ADSP.

  The creation is allowed if at least one of the following is true:
  - The user has ALTER authority to the data set through the generic profile or global access checking.
  - The user has CREATE authority in the group.

  RACF does not create a profile.
- The data set name is not covered by an existing generic profile and the user does not have ADSP.

  If PROTECTALL is not in effect, the creation is allowed, but RACF does not create a profile. See Note 2.
- The user has ADSP and the data set belongs to a group of which the user is a member.

  The creation is allowed only if the user has CREATE authority in the group. If the creation is allowed, RACF creates a discrete profile for the data set.
- The REQUEST=DEFINE preprocessing exit routine allows RACF protection.
- The user has the OPERATIONS attribute except when both of the following are true:
  1. The user is connected to the group with less than CREATE authority.
  2. The user has less than ALTER access to the data set if it protected by a generic profile.

  If the user has the group-OPERATIONS attribute (that is, the user is connected to a superior group with the OPERATIONS attribute), the group for which the new data set is being created must be within the scope of that superior group.

If PROTECTALL is not in effect, any user without ADSP can create a data set whose high-level qualifier is neither a RACF user ID (user data set) nor a RACF group name (group data set), but the data set cannot be RACF-protected. Note that a dummy group (a group that has no users connected to it) can be defined for the high-level qualifier of these data sets so that they can then be RACF-protected.

Notes:

1. In all cases, if the user specifies the PROTECT=YES or SECMODEL parameter on the JCL DD statement, or the PROTECT or SECMODEL operand on the TSO ALLOCATE command (these operands request that RACF create a discrete profile), RACF treats the user the same as a user with ADSP. However, because the use of these operands is voluntary, an installation cannot use the operands to control the creation of data sets.

2. If PROTECTALL is in effect at your installation, a user cannot create a new data set unless the data set is RACF-protected by either a discrete or generic profile. However, instead of rejecting all creation requests for unprotected data sets, PROTECTALL also allows installations to issue warning messages. For more information on the PROTECTALL option, see “RACF-Protecting All Data Sets (PROTECTALL Option)” on page 123.

Data Set Profile Ownership

Each data set profile defined to RACF requires a RACF-defined user or group as the owner of the profile. The owner (if a user) has full control over the profile, including the access list.

If the owner of the data set profile is a group, users with group-SPECIAL in that group have full control over the profile.

Ownership of data set profiles is assigned when the profiles are defined to RACF. Note that ownership of a data set profile does not mean that the owner can automatically access that data set. To access a data set, the owner must still be authorized in the profile’s access list, unless the high-level qualifier of the profile name is the owner’s user ID.

In some cases, the OWNER field of a discrete data set profile can be changed simply by renaming the data set. For details, see *z/OS SecureWay Security Server RACF System Programmer’s Guide*.

Data Set Profiles

The following topics describe what occurs when data sets are protected by profiles.

Protection Through Discrete Profiles

Users can protect data sets with discrete profiles in the following ways:

- Automatically when they create a permanent data set, if they have the ADSP attribute and ADSP is active on the system
- When they specify the PROTECT or SECMODEL parameter on a JCL DD statement for a new data set, or the PROTECT or SECMODEL operand on the TSO ALLOCATE command for a new permanent DASD data set
- When they issue the ADDSD command with the SET operand for permanent existing data sets

Two steps occur when a user defines a data set with a discrete profile. Only when RACF has completed both of the following steps is the data set protected:

1. RACF sets an indicator to notify the system that the data set is RACF-protected. This condition is called *RACF-indicated*.

   The indicator is in the DSCB for a non-VSAM DASD data set and in the catalog entry for a VSAM data set. The indicator for a tape data set is in the tape volume profile for the volume that contains the data set.

   Note: See *z/OS SecureWay Security Server RACF System Programmer’s Guide* for information on moving RACF-indicated data sets to other systems and using utilities with RACF-protected data sets.

2. RACF adds the discrete profile to the RACF database.

   For tape data sets, RACF also creates a discrete tape volume profile, unless a tape volume profile already exists for the volume or the TAPEVOL class is not active.

Notes:

1. Scratching a DASD data set that is RACF-protected with a discrete profile causes RACF to delete the data set profile from the RACF database.
2. Specifying DISP=DELETE for a tape data set only causes the data set to be uncataloged if it was cataloged; it does not remove RACF protection from the data set.

Protection through Generic Profiles

By using generic profiles, your installation can reduce both the number of profiles required to protect data sets and the size of the RACF database, thus making RACF protection easier to administer. In addition, generic profiles are loaded into storage when first needed, are not deleted when the data set they protect is deleted, and are not volume-specific (that is, data sets protected by a generic profile can reside on any volume).

You can define a generic profile to protect data sets in one of the following ways:

- By issuing the ADDSD command and specifying the generic characters *, %, or, if enhanced generic naming is active, ** in the profile name. Profile names that contain generic characters can protect a number of similarly named data sets.
- By issuing the ADDSD command and specifying the GENERIC operand. Use this operand when the profile name you specify does not contain any generic characters, in which case it is a fully qualified generic profile. A fully qualified generic profile protects only those data sets whose name matches the profile name exactly. For example, you might define a fully qualified generic profile to protect data sets with the same name that reside on different volumes.

Rules for Generic Data Set Profile Names

When You Can Specify Generic Profile Names

You can create a profile with a generic name when either of the following is true for the class of the profile:

- The SETROPTS GENERIC(DATASET) option is in effect. Not only does this option allow the creation of generic profiles, it also causes RACF to use generic profiles during authorization checking.
- The SETROPTS GENCMD(DATASET) option is in effect. In this case, generic profiles can be created and modified, but RACF does not use them during authorization checking. This is intended for use when migrating from discrete profiles to generic profiles.

Some of the rules for generic characters are different between general resource and data set generic profiles. For more information, see “Rules for Generic Profile Names” on page 200 and *z/OS SecureWay Security Server RACF Command Language Reference*.

The following rules apply to generic data set profile names:

- Valid generic characters are *, %, and **:
  - Specify % in the profile name to match any single non-blank character (except a period) in the same position of the resource name.
  - Specify * or ** in the profile name to match more than one character in the same position of the resource name. For data set profile names, you can specify ** only if the SETROPTS EGN option is in effect. For a complete description, with examples, of how to specify * and **, see *z/OS SecureWay Security Server RACF Command Language Reference*.

- For profiles in the DATASET class, the high-level qualifier of the profile name can neither contain nor be a generic character. Here are some examples:

  ABC.EF* Valid

  ABC.EF.** Valid

  A%C.EFG Invalid

  ***.EFG** Invalid

  ABC*.XYZ Invalid

  ****.XYZ** Invalid

  Note: You may see data set names with the high-level qualifiers of &&TEMP and **SYSUT. These data sets are created internally by the IEHMOVE program and should not be used for any other reason.

  RACF enforces the rule that data set qualifiers can be no longer than eight characters. Therefore, in generic data set profiles, the generic characters * and ** cannot be used to match qualifiers that are longer than eight characters.

When to Do a Generic Refresh

After you define or change generic profiles, activate your changes by entering:

```
SETROPTS GENERIC(classname) REFRESH
```

Also, changes to RACFVARS variables require a generic refresh on all classes that use those variables.

Choosing between Discrete and Generic Data Set Profiles

When you create a profile in the DATASET class, you can create either a discrete or generic profile.

Choose a *generic profile* for the following reasons:

- If you want to protect more than one data set with the same security requirements. The data sets protected by a generic profile must have some identical characters in their names. The profile name contains one or more generic characters (*, **, or %).
- If you have a single data set that might be deleted, then re-created, and you want the protection to remain the same, you can create a fully qualified generic profile. The name of a fully qualified generic profile matches the name of the data set it protects. Unlike a discrete profile, a fully qualified generic profile is not deleted when the data set itself is deleted.

Choose a *discrete profile* for the following reasons:

- To protect one data set that has unique security requirements. The name of a discrete profile matches the name of the data set it protects.
- To allow changes to a data set profile to take effect immediately, without needing to refresh in-storage copies of the profile.

Notes:

1. All of the members of a partitioned data set are protected by one profile, the profile that protects the data set.
2. All of the components of a VSAM data set are protected by one profile, the profile that protects the cluster name. You do not need to create profiles that protect the index and data components of a cluster.

3. For a generic profile, unit and volume information is ignored because the data sets that are protected under the generic profile can reside on many different volumes.

Generic Profile Checking for the DATASET Class

The rules for access-authorization checking of generic profiles for data sets are as follows:

- Generic profiles are not checked unless generic profile checking is active for the DATASET class. To activate it, enter:

  ```
  SETROPTS GENERIC(DATASET)
  ```

- If generic profile checking is in effect for the DATASET class, RACF examines the profiles as follows:
  - For a discrete profile (if the caller indicates that the data set is RACF-indicated).
  - For a fully qualified generic profile.
  - For other generic profiles in the order of *most specific to least specific* profile name. See Table 10 and Table 11 on page 159.
- After a profile is found, RACF uses information in the profile to do authorization checking. For a complete description, see “Authorization Checking for RACF-Protected Resources” on page 633.

If the data set is RACF-indicated, RACF first checks for a discrete profile. If a discrete profile does not exist, RACF examines the generic profiles in the order of *most specific to least specific* profile name. Therefore, if a discrete profile does not exist, RACF uses the most specific matching generic profile.

If the data set is *not* RACF-indicated, RACF examines the generic profiles in the order of *most specific to least specific,* and uses the most specific matching generic profile.

Note: To determine which generic profile is the most specific match to a particular data set name, you can use the LISTDSD command with the GENERIC option.

Table 10 and Table 11 on page 159 list some generic profiles from the DATASET class. This figure represents the order in which RACF checks the generic profiles when it performs access-authorization checking. (This order is also the order that RACF commands such as SEARCH would list these generic profiles.)

Table 10. Sample Data Set Profile Names in Order from Most Specific to Least Specific (EGN Off)

		Data Sets Being Accessed		
Profile Name	**Profile Type**	**SALES.DATA**	**SALES.DATA.TEST**	**SALES.YEARLY.QUOTA**
SALES.A	Fully qualified generic			
SALES.DATA	Discrete	X		
SALES.DATA	Fully qualified generic	X		
SALES.DATA.%	Generic			
SALES.DATA.*	Generic		X	
SALES.DATA%	Generic			
SALES.DATA*	Generic	X	X	

Table 10. Sample Data Set Profile Names in Order from Most Specific to Least Specific (EGN Off) (continued)

		Data Sets Being Accessed		
Profile Name	**Profile Type**	**SALES.DATA**	**SALES.DATA.TEST**	**SALES.YEARLY.QUOTA**
SALES.DAT%	Generic	X		
SALES.DAT*	Generic	X	X	
SALES.DISK.*	Generic			
SALES.YEARLY.QUOTA	Discrete			X
SALES.YEARLY.QUOTA	Fully qualified generic			X
SALES.YEARLY.*	Generic			X
SALES.%ATA	Generic	X		
SALES.*.QUOTA	Generic			X
SALES.*.QUOTA*	Generic			X
SALES.*	Generic	X	X	X

Note: RACF ignores a discrete profile if a data set is not RACF-indicated. Any data set that has a discrete profile must be RACF-indicated.

Table 11. Sample Data Set Profile Names in Order from Most Specific to Least Specific (EGN On)

		Data Sets Being Accessed		
Profile Name	**Profile Type**	**SALES.DATA**	**SALES.DATA.TEST**	**SALES.YEARLY.QUOTA**
SALES.A	Fully qualified generic			
SALES.DATA	Discrete	X		
SALES.DATA	Fully qualified generic	X		
SALES.DATA.%	Generic			
SALES.DATA.*	Generic		X	
SALES.DATA.**	Generic	X	X	
SALES.DATA%	Generic			
SALES.DATA*	Generic	X		
SALES.DAT%	Generic	X		
SALES.DAT*	Generic	X		
SALES.DISK.*	Generic			
SALES.YEARLY.QUOTA	Discrete			X
SALES.YEARLY.QUOTA	Fully qualified generic			X
SALES.YEARLY.*	Generic			X
SALES.%ATA	Generic	X		
SALES.*	Generic	X		
SALES.*.QUOTA	Generic			X
SALES.*.QUOTA*	Generic			X
SALES.**.DATA	Generic	X		
SALES.**.QUOTA	Generic			X
SALES.**	Generic	X	X	X

Table 11. Sample Data Set Profile Names in Order from Most Specific to Least Specific (EGN On) (continued)

		Data Sets Being Accessed		
Profile Name	**Profile Type**	**SALES.DATA**	**SALES.DATA.TEST**	**SALES.YEARLY.QUOTA**

Note: RACF ignores a discrete profile if a data set is not RACF-indicated. Any data set that has a discrete profile must be RACF-indicated.

To find out which profiles could protect a data set, take the following steps:

1. Find out if there is a discrete profile protecting the data set:

   ```
   LISTDSD DATASET('dataset-name')
   ```

   If a discrete profile exists for the data set, this command lists the contents of the profile.
2. If no discrete profile protects the data set, issue the LISTDSD command again with the GENERIC option:

   ```
   LISTDSD DATASET('dataset-name') GENERIC
   ```

   If a generic profile exists for the data set, this command lists the contents of the profile.
3. There may well be other generic profiles that have the potential to protect the data set. These profiles are listed by the SEARCH command in the order that RACF would use them. Because all data set profiles begin with a user ID or group name, you can use the FILTER operand to show only those profiles that could protect a data set, as shown in the following examples:

   ```
   SEARCH CLASS(DATASET) FILTER(userid.**)
   SEARCH CLASS(DATASET) FILTER(groupname.**)
   ```

To see which of two generic profiles is more specific, compare the profile names, character by character. Where they first differ, if one has a discrete character and the other has a generic character, the one with the discrete character wins. If both have a generic character where they differ, then:

- If one has a % and the other has a * or **, the one with % wins.
- If one has a * and the other has a **, the one with * wins.

Note: If two profile names fit except for one character position, the following is the order in which RACF examines them:

```
blank
.
$  (X'5B')
#  (X'7B')
@  (X'7C')
A through Z
0 through 9
%
*
```

Attention

These characters—**$, #, and** @—may be displayed differently on terminals outside the United States. Therefore, use the characters with the hexadecimal equivalents shown above.

For example, the following profile names all fit in the first three character positions (A.B), and are shown in the order RACF examines them:

```
A.B
A.B.B
A.BA
A.BZ
A.B0
A.B9
A.B%
A.B*
```

When in doubt about the search order, create sample profiles and check the order of profile names shown by the SEARCH command.

Using SETROPTS PROTECTALL and SETROPTS GENERIC(DATASET) Together

If PROTECTALL is in effect at your installation, generic profile checking should also be in effect. This allows you to create or access a data set if one of the following conditions is met:

- The data set is protected by a discrete profile.
- The data set is protected by a generic profile.
- The access is allowed by global access checking.

For users with alter authority, RACF allows renaming a data set from a name covered by a global entry to another name covered by a global entry. Similarly, renaming is allowed from a name covered by one generic profile to a name covered by another generic profile. Renaming is not allowed from a name covered by a generic profile to one covered by a global entry, because this could allow the user to remove protection from the data set.

If PROTECTALL is in effect and generic profile checking is not, only users who have ADSP or specify PROTECT=YES can create new data sets.

After defining, altering, or deleting a generic profile, the following command ensures that the profile is in effect during authorization checking:

```
SETROPTS GENERIC(DATASET) REFRESH
```

RACF is invoked whenever a data set is accessed (whether or not the data set is RACF-indicated) and whenever DASD space is allocated for a data set (whether or not the user has the ADSP attribute or has specified PROTECT=YES on the JCL statement). When RACF is invoked for a data set that is not RACF-indicated, RACF checks only predefined generic profiles and the global access checking table. If PROTECTALL is not in effect and RACF cannot find an appropriate generic profile or a matching entry in the global access checking table, RACF accepts the access request by default.

Attention

Data sets that are not RACF-indicated but are protected by a generic profile are *not protected* if they are transferred (in any way) or available (such as through shared DASD) to another system that does not have RACF and appropriate predefined generic profiles.

Authority to Modify Generic Profiles

To modify a generic profile, a user must be the profile owner, or have the SPECIAL (or group-SPECIAL, if applicable) attribute, or have a user ID identical to the profile's high-level qualifier. Unless one of these conditions is met, the user cannot alter the generic profile, even if the user has ALTER access authority to the profile. Note that the access list in a generic profile does not apply to the profile itself. See

z/OS SecureWay Security Server RACF Command Language Reference for descriptions of the authorities needed to issue particular RACF commands.

Conditional Access Lists for Data Set Profiles

RACF allows installations to specify conditional access lists for data sets. You can require that a user or job enter the system from a particular device when accessing data sets. Specifically, you can do the following:

- By specifying WHEN(TERMINAL(...)) on the PERMIT command, you can require that a user be logged onto a particular terminal.

  For this support to take effect, the TERMINAL class must be active.
- By specifying WHEN(CONSOLE(...)) on the PERMIT command, you can require that a user be logged onto a particular console.

  For this support to take effect, the CONSOLE class must be active.
- By specifying WHEN(JESINPUT(...)) on the PERMIT command, you can require that the batch job accessing the data set has been submitted from a particular JES input device.

  For this support to take effect, the JESINPUT class must be active.
- By specifying WHEN(APPCPORT(...)) on the PERMIT command, you can require that a user enter the system from a particular partner LU.

  For this support to take effect, the APPCPORT class must be active.

Note: If an access list contains more than one condition, *any* of the conditions allows the specified access. For example, if you enter the PERMIT command with WHEN(CONSOLE(01) TERMINAL(20)) specified, you allow the access when *either* console 01 *or* terminal 20 is used.

Universal Access Authority (UACC) for Data Sets

Each data set profile you define with RACF requires a universal access authority (UACC). The UACC is the default access authority that RACF gives to users and groups that are not defined in the profile's access list. If one of these users or groups requests access to a data set that is protected by the profile, RACF grants or denies the request based on the UACC. UACC coverage also extends to users that are not defined to RACF and batch jobs that are not associated with a RACF-defined user. A batch job has no user ID associated with it in the following cases:

- There is no user ID propagation in the system, and no user ID or password was specified.
- The release of JES that is installed does not support user ID propagation.
- The job originated from an NJE, RJE, or card reader, and no USER parameter was specified on the JOB statement.

In some cases, jobs originating from NJE can have a user ID, depending on the NODES class profiles that are defined on your system.

If you specifically assign an access authority to a user or group, the authority you specify overrides the UACC assigned to the data set. Also, if the access checking defined in the global access checking table is higher than the UACC assigned to the data set, the entry in the global access checking table overrides the UACC.

For a given data set:

- If you set UACC to NONE, all users are refused access to the data set because they are not authorized to access the data set through an access list, global access checking, the OPERATIONS attribute, or the WARNING indicator.

- If you set UACC to READ, EXECUTE, UPDATE, CONTROL, or ALTER, all users can access the data set at the specified level of authority, unless they are specifically excluded by security classification checking or an entry in the standard access list, or the user ID has the RESTRICTED attribute.

Note: If you have users who are not defined to RACF, you can use ID(*) instead of UACC to ensure that only RACF-defined users access the resource. The following examples illustrate the difference between UACC(READ) and ID(*) ACCESS(READ).

- To allow *all* users on the system to use a data set, specify UACC(READ) for the profile, as follows:

  ```
  RDEFINE profile-name UACC(READ)
  ```

- To allow *only RACF-defined* users on the system to use a data set, specify UACC(NONE) for the profile, and then issue the PERMIT command with ID(*) and ACCESS(READ) specified:

  ```
  RDEFINE profile-name UACC(NONE)

  PERMIT profile-name ID(*) ACCESS(READ)
  ```

Automatic Profile Modeling for Data Sets

You can set up automatic modeling for new data set profiles (whether discrete or generic). You can do this for:

- Data set profiles for selected users
- Data set profiles for selected groups
- GDG data sets

Automatic Profile Modeling for User Data Set Profiles

You can specify a model data set profile to be used whenever new user data set profiles are created for a specific user. Information from the model profile is copied to any data set profile with the specified user ID as high-level qualifier.

To do this, follow these steps:

1. For each user for which modeling is to be done, specify the profile that is to be used as a model:

   ```
   ALTUSER userid MODEL(model-profile-name)
   or
   ADDUSER userid MODEL(model-profile-name)
   ```

   Note: When specifying the MODEL operand, do *not* specify the user's user ID on the model profile name.

2. If necessary, create a model data set profile:

   ```
   ADDSD 'userid.model-profile-name' MODEL
         ...other appropriate operands such as UACC and AUDIT...

   PERMIT 'userid.model-profile-name' ID(appropriate-users-or-groups)
          ACCESS(access-authority)
   ```

   Notes:

   a. With the MODEL operand specified, no actual data set need exist with the specified profile name. A generic profile cannot be a model profile.

   b. A profile created with the MODEL operand is not intended to actually protect a data set (and does not cause an existing data set to be RACF-indicated). However, if a data set with the same name exists, the model profile might be used to protect that data set. Therefore, IBM recommends that you choose a profile name such that the profile does not match any data sets.

3. When you are ready to start using model profiles for user data sets, issue the SETROPTS command with MODEL(USER) specified:

   ```
   SETROPTS MODEL(USER)
   ```

4. After the SETROPTS command has been issued, if a user creates a user data set profile for another user, and that profile had the MODEL operand specified, information from the model profile is *always* copied into the new user data set profile.

Example:

The following commands set up a model profile named SUE.SAMPMOD for user SUE. The model specifies a UACC of NONE and gives READ access to SAM, JOE, and GROUP1:

```
(1) ALTUSER SUE MODEL(SAMPMOD)
(2) ADDSD 'SUE.SAMPMOD' MODEL UACC(NONE)
(3) PERMIT 'SUE.SAMPMOD' ID(SAM JOE GROUP1) ACCESS(READ)
(4) SETROPTS MODEL(USER)
```

User SUE then issues the following command:

```
(5) ADDSD 'SUE.DATA' UACC(READ)
```

In this example:

(1) indicates to RACF that automatic profile modeling is to be used for new profiles beginning with SUE.

(2) creates a profile named SUE.SAMPMOD. With the MODEL operand specified, no actual data set named SUE.SAMPMOD needs to exist. However, if a data set named SUE.SAMPMOD does exist, it is protected by the profile named SUE.SAMPMOD.

(3) specifies an access list for profile SUE.SAMPMOD.

(4) turns on automatic profile modeling for all of the users who have the MODEL operand set in their user profiles.

(5) creates profile SUE.DATA with UACC(READ). RACF copies the access list from SUE.SAMPMOD (SAM, JOE, and GROUP1 have READ access). With UACC(READ) specified on the ADDSD command, the UACC(NONE) value from SUE.SAMPMOD is not used. Note that the copied information can be changed during the copy. See “Possible Changes to Copied Profiles When Modeling Occurs” on page 40.

Automatic Profile Modeling for Group Data Set Profiles

You can specify a model data set profile to be used whenever new group data set profiles are created in a specific group. Information from the model profile is copied to any data set profile with the specified group name as high-level qualifier.

To do this, take the following steps:

1. For each group for which modeling is to be done, specify the profile to be used as a model by entering one of the following commands:

   ```
   ALTGROUP groupname MODEL(model-profile-name)
   ADDGROUP groupname MODEL(model-profile-name)
   ```

   Note: When specifying the MODEL operand, do *not* specify the group’s group name on the model profile name.

2. If necessary, create a model data set profile:

```
ADDSD 'groupname.model-profile-name' MODEL
      ...other appropriate operands such as UACC and AUDIT...

PERMIT 'groupname.model-profile-name' ID(appropriate-users-or-groups)
       ACCESS(access-authority)
```

Notes:

a. With the MODEL operand specified, no actual data set need exist with the specified profile name.

b. A profile created with the MODEL operand is not intended to actually protect a data set (and does not cause an existing data set to be RACF-indicated). However, if a data set with the same name exists, the model profile might be used to protect that data set. Therefore, IBM recommends that you choose a profile name such that the profile does not protect any data sets.

3. When you are ready to start using model profiles for group data sets, issue the SETROPTS command with MODEL(GROUP) specified:

   ```
   SETROPTS MODEL(GROUP)
   ```

4. After the SETROPTS command has been issued, if a user creates a group data set profile for a group for which the MODEL operand has been specified, information from the model profile is *always* copied into the new group data set profile.

Example:

The following commands set up a model profile named GROUP1.SAMPMOD for group GROUP1. The model specifies a UACC of NONE and gives READ access to SAM, JOE, and GROUP1:

```
(1) ALTGROUP GROUP1 MODEL(SAMPMOD)
(2) ADDSD 'GROUP1.SAMPMOD' MODEL UACC(NONE)
(3) PERMIT 'GROUP1.SAMPMOD' ID(SAM JOE GROUP1) ACCESS(READ)
(4) SETROPTS MODEL(GROUP)
```

A user then issues the following command:

```
(5) ADDSD 'GROUP1.DATA' UACC(READ)
```

In this example:

(1) indicates to RACF that automatic profile modeling is to be used for new profiles beginning with GROUP1.

(2) creates a profile named GROUP1.SAMPMOD. With the MODEL operand specified, no actual data set named GROUP1.SAMPMOD needs to exist. However, if a data set named GROUP1.SAMPMOD does exist, it is protected by the profile named GROUP1.SAMPMOD.

(3) specifies an access list for profile GROUP1.SAMPMOD.

(4) turns on automatic profile modeling for all groups that have the MODEL operand set in their group profiles.

(5) creates profile GROUP1.DATA with UACC(READ). RACF copies the access list from GROUP1.SAMPMOD (SAM, JOE, and GROUP1 have READ access). With UACC(READ) specified on the ADDSD command, the UACC(NONE) from GROUP1.SAMPMOD is not used. Note that the copied information can be changed during the copy. See “Possible Changes to Copied Profiles When Modeling Occurs” on page 40.

Automatic Profile Modeling for GDG Data Sets

You can use automatic profile modeling for GDG data sets. For more information, see “Protecting GDG Data Sets” on page 166.

Password-Protected Data Sets

When a data set is both password-protected and RACF-protected, access to the data set is authorized through RACF authorization checking. If an authorization request for a password-protected data set is satisfied by a RACF global access table entry or a RACF data set profile, password checking is ignored.

When a data set is password-protected but not RACF-protected, access to the data set is authorized through password protection.

When a RACF-protected data set is moved to a system without RACF support, you cannot perform authorization checking. Therefore, after you have installed RACF, your users may need to maintain password protection only for those data sets that:

- Are not RACF-protected
- Are RACF-protected and are used on other systems that do not have RACF support

Password protection is not used for SMS-managed data sets. Therefore, if your installation has procedures that use password protection for data sets, you must modify these procedures accordingly.

Protecting GDG Data Sets

You can RACF-protect GDG (generation data group) data sets in one of the following ways:

- You can define a generic profile to protect all members of a GDG. This is the method that IBM recommends and it is the same as the method for protecting non-GDG data sets with a generic profile. For example, a profile of the form GDG.*basename** protects all members of a GDG and the base entry for the GDG in the catalog.

  Note that, if enhanced generic naming is in effect, a profile of the form GDG.*basename.*** provides the same protection.

  Table 12 shows examples of generic profiles that you can define to protect GDG data sets.

Table 12. Protecting GDG Data Sets Using Generic Profiles

Generic Profile Name	EGN	Protected GDG Names
`GDG.BASENAME*`	Off	`GDG.BASENAME` `GDG.BASENAME.G0123V00`
`GDG.BASENAME.**`	On	`GDG.BASENAME` `GDG.BASENAME.G0123V00`

Note: For GDG profiles, with enhanced generic naming active, you can no longer define a profile name such as GDG.ABCDEFGH* whose last qualifier contains an asterisk as the ninth character. Externally, an existing profile name of this format is shown as GDG.ABCDEFGH.**. Internally, no conversion is required because the two names are equivalent. However, you should examine existing CLISTs that generate commands to ensure that any profile names that appear in those commands are in the correct format.

- You can define discrete profiles to protect GDG data sets in the same way that you define discrete profiles to protect non-GDG data sets.

Note: Catalog management also checks authority to the GDG base name. You should create a discrete profile for the GDG base with the unit and volume of the catalog on which the GDG base resides. This protects the GDG for catalog and uncatalog functions.

- You can use the MODEL(GDG) operand on the SETROPTS command to specify that each member of a GDG can use a common profile identified by the GDG base name. The owner of the GDG data set can establish a base (index) name profile containing an access list that is accessible by all related users and groups. When MODEL(GDG) is in effect and REQUEST=AUTH processes a RACF-indicated GDG data set, RACF first looks for a profile with the base name, and, if one exists, uses this common profile.

  If you want individual access lists, do not create the profile for the base name. If the GDG base name is not defined in the RACF database, RACF uses the profile for the individual GDG name (which is the same as the RACF-processing for non-GDG data sets).

  Notes:

  1. To use GDG modeling, each generation must be RACF-indicated.
  2. Catalog management also checks authority to the GDG base name. You should create a discrete profile for the GDG base with the unit and volume of the catalog on which the GDG base resides. This protects the GDG for catalog and uncatalog functions.

Protecting Data Sets That Have Duplicate Names

You can use a fully qualified generic profile to protect data sets with the same name that reside on different volumes.

Alternatively, you can use separate, discrete profiles to define data sets having the same name. Support for data sets with duplicate names allows authorized users to:

- Move and copy RACF-protected data sets from one volume to another (for example, with the IEHMOVE system utility)
- Establish separate discrete profiles (including the access list and statistics and logging options) for data sets having the same name
- Protect data sets that have the same name and reside on non-shared volumes (such as SYS1.LINKLIB) on a loosely-coupled system that uses a shared RACF data set

RACF differentiates between data sets having the same name by examining the volume serial number of each separately protected data set.

Non-VSAM DASD Data Sets

For non-VSAM data sets, RACF uses the serial number of the volume on which the data set resides.

VSAM Data Sets

For VSAM data sets, RACF uses the volume serial number of the catalog for the VSAM or integrated catalog facility in which the data set is cataloged. If multiple catalogs for the integrated catalog facility reside on the same volume and contain entries for duplicate VSAM data sets, only one of the data sets can be protected by a discrete profile.

Tape Data Sets

If TAPEDSN is active and RACF is maintaining a TVTOC for the tape volume (TAPEVOL is active), RACF does not permit duplicate data set names on the same

volume, or on different volumes if the volumes are part of the same multivolume data set group. This restriction applies even if the data sets are not protected by RACF.

Disallowing Duplicate Names for Data Set Profiles

You can prevent identically named data sets from being defined to RACF with separate, discrete profiles by modifying the installation-replaceable module ICHSECOP. For information on modifying ICHSECOP, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

If you disallow duplicate data set profile names, data sets with the same name must be defined to RACF for protection in one common discrete profile with multiple volume serials. In this case, a data set shares the data set profile (including the access list, and statistics and logging options) with other data sets that have the same name.

Note that a fully qualified generic profile can also be used to protect data sets with identical names, regardless of which volumes they reside on.

Using the PROTECT Operand or SECMODEL for Non-VSAM Data Sets

To create a discrete profile for a new tape or non-VSAM DASD data set (if you do not have the ADSP attribute), specify the PROTECT or SECMODEL parameter on the JCL DD statement that identifies the data set (or for DASD data sets, the PROTECT or SECMODEL operand on the TSO ALLOCATE command). Note that the normal reason for a user to use PROTECT or SECMODEL instead of ADSP is that most of the user's data sets do not require discrete profiles because they are covered by generic profiles.

Protecting Multivolume Data Sets with Discrete Profiles

You can protect a multivolume data set with either a discrete or a generic profile. If a generic profile protects the data set, the fact that the data set is multivolume is irrelevant.

To create a discrete profile for a multivolume tape or non-VSAM DASD data set, you must define each volume of the data set to RACF. RACF stores the volume serial numbers in the data set's profile. When the data set is extended to another volume or deleted from a volume, that volume's serial number is automatically added to or deleted from the data set profile.

Note: You cannot rename a multivolume non-VSAM DASD data set that is protected by a discrete profile. If the data set is protected with a generic profile, it can be renamed if the new name is also covered by a generic profile.

For more information on handling multivolume data sets in addition to the following considerations, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Non-VSAM DASD Data Set Considerations

When a multivolume physical sequential DASD data set is opened for input, RACF does not require that each volume on which the data set resides be defined in the data set profile.

When an existing multivolume physical sequential data set is opened for output, a RACF-protection consistency check is performed. All volumes of the data set that

are processed by end-of-volume (when a volume switch occurs) must indicate the same RACF-protection status as the first volume opened. That is, if the first volume is RACF-protected (the DSCB indicator is on and the volume is defined in the data set profile), succeeding volumes must be RACF-protected as part of the same profile. If the first volume is not RACF-protected, succeeding volumes must not be RACF-protected.

For multivolume non-physical sequential DASD data sets, RACF performs authorization checking for each volume on which the data set resides.

Tape Data Set Considerations

When an existing multivolume data set is opened for output, a RACF-consistency check is performed. All volumes of the data set must be RACF-protected, or all volumes of the data set must not be RACF-protected.

When a volume label is changed (destroyed) during a volume label rewrite, a REQUEST=DEFINE,TYPE=DELETE is issued for the old volume serial number.

VSAM Data Set Considerations

For VSAM data sets, extending the data set to a new volume causes RACF to protect the new volume, even though RACF does not add the serial number to the data set profile. The profile for VSAM data sets contains only the volume serial number of the catalog entry for the data set.

Setting ADDCREATOR/NOADDCREATOR Options for Both DASD and Tape

When specified in a generic profile, ALTER allows users to create new data sets that are covered by that profile. If the SETROPTS NOADDCREATOR option is in effect, the user who created the profile is not automatically added to the profile's access list. Even when the SETROPTS NOADDCREATOR option is in effect, when discrete DATASET or TAPEVOL profiles have been created using RACROUTE REQUEST=DEFINE (including RACDEF), the profile creator's ID is automatically added to the list. For more information, see "Automatic Addition of Creator's User ID to Access List" on page 142.

Protecting DASD Data Sets

This section gives additional information that applies to data sets on DASD. You should already be familiar with the information contained in "Protecting Data Sets" on page 150.

Access Authorities for DASD Data Sets

You permit users and groups to access a RACF-protected data set by:

- Adding them to the access list of the discrete or generic profile that applies to the data set and
- Giving them one of the access authorities described in Table 13 on page 170.

Table 13 on page 170 describes the access authorities associated with data set profiles. Many operations for cataloged data sets involve access not only to the data set profile protecting the data set, but also to the catalog in which the data set is cataloged. For access authorities required by users who are creating, deleting, or renaming data sets, see "Controlling the Creation of New Data Sets" on page 153. For more information about authorizing users to perform data set and catalog operations with protected catalogs, see the following publications:

- *z/OS DFSMS Access Method Services*

Data Sets

- *z/OS DFSMS: Using Data Sets*
- *z/OS DFSMS: Managing Catalogs*

Table 13. Access Authorities for DASD Data Sets

NONE	Does not allow users to access the data set.
EXECUTE	For a private load library, allows users to load and execute, but not read or copy, programs (load modules) in the library. **Note:** For more information about EXECUTE authority, see "Defining Protected Program Libraries (Including Execute-Control)" on page 254. **Attention** Anyone who has READ, UPDATE, CONTROL, or ALTER authority to a protected data set can create a copy of it. As owner of the copied data set, that user has control of the security characteristics of the copied data set and can downgrade it. For this reason, you should assign a UACC of NONE, and then selectively permit a small number of users to access your data set, as their needs become known. (For information on how to permit selected users or groups to access a data set, see *z/OS SecureWay Security Server RACF General User's Guide*.)
READ	Allows users to access the data set for reading only. (Note that users who can read the data set can copy or print it.)
UPDATE	Allows users to read from, copy from, or write to the data set. However, UPDATE does not authorize a user to delete, rename, move, or scratch the data set. Allows users to perform normal VSAM I/O (not improved control interval processing) to VSAM data sets.
CONTROL	For VSAM data sets, is equivalent to the VSAM CONTROL password. That is, it allows users to perform improved control interval processing. This is control-interval access (access to individual VSAM data blocks), and the ability to retrieve, update, insert, or delete records in the specified data set. For non-VSAM data sets, CONTROL is equivalent to UPDATE.
ALTER	Allows users to read, update, delete, rename, move, or scratch the data set. When specified in a discrete profile, ALTER allows users to read, alter, and delete the profile itself *including the access list*. **Note:** ALTER does not allow users to change the owner of the profile using the ALTDSD command. However, if a user with ALTER access authority to a discrete data set profile renames the data set, changing the high-level qualifier to his or her own user ID, then both the data set and the profile are renamed, *and* the OWNER of the profile is changed to the new user ID. When specified in a generic profile, ALTER gives users *no* authority over the profile itself. When specified in a generic profile, ALTER allows users to create new data sets that are covered by that profile.

Suggestions for Assigning Access Authorities to DASD Data Sets

When protecting catalogs, be sure that users and groups have a sufficient level of access authority to each protected entity along the path to a data set that they are required to access.

The level of RACF access authority that a user or group requires to perform operations on VSAM data sets or catalogs is similar to the level of authorization required when passwords are used. For more information, see “Comparison of Password and RACF Authorization Requirements for VSAM” on page 172.

For a discussion of the levels of RACF access authority required to perform operations against catalogs, see *z/OS DFSMS: Managing Catalogs*.

Erasing of Scratched (Deleted) DASD Data Sets

Installations can control the erasure of security-sensitive data set extents with the ERASE operand (erase-on-scratch) on the SETROPTS command. If a DASD data set profile has the erase indicator set, ERASE specifies that data management is to erase (overwrite) the contents of any scratched or released data set extents that are part of a DASD data set protected by that profile.

Users can set the erase indicator in both discrete and generic profiles by using the ADDSD and ALTDSD commands. Users can specify erasure for both single volume and multiple volume DASD data sets. However, to have the data set erased when scratched, the installation must also activate erase-on-scratch with the SETROPTS command.

The SETROPTS command has several options on the ERASE operand that allow an installation to override user specifications. These erase-on-scratch options allow an installation to:

- Specify that *all* DASD data set extents are always erased when scratched or released, regardless of the erase indicator in the profile. (This includes temporary data sets.) When this option is selected, installation exit routines *cannot* prevent any data set from being erased by overriding this option.
- Specify a security level (SECLEVEL) at which all data sets at this security level or higher are always erased when scratched or released, regardless of the erase indicator in the profile.
- Specify that only data sets that have the erase indicator in their profiles are erased when scratched or released.
- Specify that no data sets under RACF control are erased when scratched or released.

Notes:

1. RACF does not perform the actual erasure, but maintains an indicator for data management.
2. If you have not specified that you want *all* data sets erased, you can still provide for the erasing of sensitive temporary data sets by using the naming conventions table or RACF exit routines to conditionally convert the temporary data set names to the form of a permanent name that is covered by a profile that specifies erase-on-scratch.
3. In addition to the RACF-controlled erasure, any VSAM data set with a catalog entry that specifies erase is erased.

Comparison of Password and RACF Authorization Requirements for VSAM

The RACF authorization requirements are the same as the password requirements for most VSAM operations. The RACF authorization levels of ALTER, CONTROL, UPDATE, and READ correspond to the password levels of MASTER, CONTROL, UPDATE, and READ, respectively.

As an example, deleting a VSAM data set requires the MASTER-level password of either the data set or the catalog that describes the data set. Deleting a RACF-protected VSAM data set requires ALTER authorization to the data set or the catalog.

There are a few exceptions to the one-to-one correspondence of the RACF and password authorization levels. Access requirements for these exceptions can be found in the publications pertaining to the operations being performed.

Protecting Catalogs

You can protect many catalog functions including catalog locking and SMS-related functions, by creating profiles in the FACILITY class. For information on creating these profiles and authorizing users to perform catalog operations on protected catalogs, see the following publications:

- *z/OS DFSMS: Managing Catalogs*
- *z/OS DFSMS Access Method Services*
- *z/OS DFSMS: Using Data Sets*

Protecting DASD System Data Sets

When you are planning to RACF-protect system data sets, consider:

- The way in which the system uses the data set
- The way in which your users normally use the data set
- The level of protection you want for the data set

The system data sets can be divided into two categories: data sets for which RACF protection is bypassed when the system accesses them, and data sets for which RACF protection is enforced when the system accesses them.

Bypassed RACF Protection

For a data set of this type, RACF-protection is bypassed when the system accesses the data set to perform its normal system function, but is enforced when a user attempts to access the data set for normal data set operations.

For example, when the program libraries defined in LNKLST*xx* are opened during IPL, RACF protection is bypassed. A user can fetch any program stored in these libraries during IPL, but if the user attempts to open one of the libraries, RACF protection is enforced. Assuming SYS1.LINKLIB is defined in LNKLST*xx*, it can be RACF-protected giving UPDATE access authority to the system programmers who maintain the data set. You can use UACC(NONE) if you do not want anyone other than the system programmers to open the library. The UACC(NONE) specification does not prevent any user from executing any program contained in SYS1.LINKLIB, but it does prevent users from, for example, specifying SYS1.LINKLIB as part of JOBLIB, STEPLIB, or on the TSO CALL command.

Examples of other system data sets that fall into this category are:

SYS1.CMDLIB
SYS1.DUMPnn
SYS1.LOGREC
SYS1.LPALIB
SYS1.MANn
SYS1.NUCLEUS
SYS1.PARMLIB
SYS1.PROCLIB
SYS1.SVCLIB

System data sets that are frequently accessed by all users (for example, SYS1.HELP and SYS1.MACLIB) are good candidates for inclusion in a global access checking table.

Enforced RACF Protection

For a data set of this type, RACF protection is enforced when the system accesses the data set for its normal system function on behalf of a specific user. When you protect this type of data set, any user who requests the system function associated with the data set must have a sufficient level of access authority to the data set for the function to work correctly.

For example, when you RACF-protect the SYS1.BRODCAST data set, you should give all users UPDATE access authority to the data set because the TSO SEND command opens the SYS1.BRODCAST data set for update. You can give UPDATE access authority by placing 'SYS1.BRODCAST'/UPDATE in the global access checking table. The system programmers who maintain the data set can be given ALTER access authority by way of a discrete profile or a fully-qualified generic profile.

Examples of other system data sets that fall into this category are:

SYS1.HELP
SYS1.MACLIB
SYS1.PARMLIB
SYS1.SAMPLIB

Note: SYS1.PARMLIB is in both lists of examples because there are some system functions for which RACF protection is bypassed when accessing SYS1.PARMLIB, and some for which it is enforced. For example, TCAS requires access to SYS1.PARMLIB.

DASD Volume Authority

By defining profiles in the DASDVOL class, you can define DASD volumes to RACF and authorize users to perform maintenance operations (such as dump, restore, scratch, and rename) without having access to the data set profiles protecting the data sets. (If a user does not have the necessary DASDVOL authority, he or she must have the necessary authority in the DATASET class to each of the data sets on the volume.)

The access authority that you give to a user depends on the product that the user is using:

- If the user is using DFDSS (Data Facility Data Set Services), the access authority required depends on the specific action that the user is requesting (for example, DUMP with DELETE or DUMP without DELETE). For a complete description of the access authorities required, see *z/OS DFSMSdss Storage Administration Reference*.

- If the user is using the DADSM scratch function, ALTER access authority allows the user to scratch data sets on the volume.

  Note: If a data set protected by a discrete profile is scratched, the discrete profile is deleted, or, in the case of a multivolume data set, the volume serial number is removed from the data set profile.
- If the user is using the Device Support Facilities (ICKDSF) program, ALTER allows the user to rename DASD volumes.
- Other products can also check for authorization in the DASDVOL class.
- **Exception:** DASDVOL authority may or may not allow users to use the TSO or IDCAMS DELETE commands, or PDF option 3.2, for scratching or renaming. This would depend on the level of those products on the system.
- **Exception:** DASDVOL authority does not allow users to work with SMS-managed volumes. Instead, you can either give the user the OPERATIONS (or group-OPERATIONS) attribute or, if you have the necessary software, define the user as a DFDSS-authorized storage administrator. For more information on the latter alternative, see "DFDSS-Authorized Storage Administration".

As an alternative to assigning the OPERATIONS or group-OPERATIONS attribute, DASDVOL authority allows you to authorize operations personnel to access only those volumes that they must maintain. Using DASDVOL authority is also more efficient for functions such as volume dumping, because only one authorization check for the volume needs to be issued, instead of individual requests for each data set on the volume. For a description of the OPERATIONS attribute, see "The OPERATIONS Attribute" on page 75.

If the volume serials do not readily allow the use of * or % as generic characters in DASDVOL profile names, consider creating profiles in the GDASDVOL class. See "Creating Resource Group Profiles" on page 221.

DFDSS-Authorized Storage Administration

Operations personnel must routinely perform maintenance operations such as copying, reorganizing, cataloging, and scratching data. To perform these operations on RACF-protected resources, they need RACF authorization. Two methods of providing this authorization are:

- Giving the user the OPERATIONS or group-OPERATIONS attribute
- Defining profiles for volumes in the DASDVOL class and authorizing the user to access the volumes

Both of these methods have certain drawbacks. For example, the OPERATIONS or group-OPERATIONS attribute may give the user more authority to look at individual data sets than you would like. Defining DASDVOL profiles may require more administrative overhead than you would like and, in addition, does not allow a user to work with SMS-managed volumes.

If you have DFDSS or DFSMS/MVS, there is a third method you can use. You can define special FACILITY-class profiles that let a user act as a DFDSS-authorized storage administrator. Once the profiles and permissions are set up, the user obtains authorization to the required volumes by specifying the ADMINISTRATOR option on the appropriate DFDSS command.

DFDSS-authorized storage administration is more flexible and requires less administrative work to maintain. For complete details on how to set up and use the support, see *z/OS DFSMSdss Storage Administration Reference*.

Protecting Data on Tape

This section gives detailed information that applies to *tape data sets* and *tape volumes*. You should already be familiar with the information in "Protecting Data Sets" on page 150, which applies to both DASD and tape data sets.

RACF allows you to establish access requirements for both tape data sets and tape volumes. To protect data on tape, you can do either or both of the following:

- Control access to the tape volume by issuing SETROPTS CLASSACT(TAPEVOL).

  The TAPEVOL class is not active when RACF is first installed.
- Control access to individual tape data sets on the tape volume by issuing SETROPTS TAPEDSN.

  NOTAPEDSN is in effect when RACF is first installed.

Choosing Which Tape-Related Options to Use

The following sections list considerations for each combination of tape volume (TAPEVOL) and tape data set (TAPEDSN) protection.

Tape Data Set and Tape Volume Protection (TAPEDSN Active and TAPEVOL Active)

- Using the ADDSD command for a tape data set results in two discrete profiles: an automatic tape volume profile that contains a tape volume table of contents (TVTOC) and a tape data set profile. This means that RACF provides:
  - Checking of the RACF security retention period (the number of days that must elapse before the data set can be deleted or overwritten).
  - Verification for the full 44-character data set names.
  - Protection for multiple data sets on a volume if all of the data sets have the same access requirements.
  - Multivolume data set protection.
  - RACF protection for the volume.
  - Automatic deletion of the data set and tape volume profiles when the data set or tape volume is overwritten and discrete protection for the data set has expired.

    Normally, you use the SET operand (which is the default) on the ADDSD command. If the tape volume and data set profiles get out of synchronization (that is, if the tape volume profile refers to a data set profile that does not exist or vice versa), use either the NOSET or SETONLY operand. Use NOSET if you have a data set profile but no tape volume profile. Use SETONLY if you have a tape volume profile but no data set profile.
  - Having ADSP or specifying PROTECT=YES on the JCL DD statement also results in two discrete profiles, just as the ADDSD command does.
  - Data management calls RACF in the DATASET class.

  For tapes being opened for input, data management issues a RACROUTE REQUEST=AUTH, CLASS=DATASET, DSTYPE=T macro. For tapes being opened for output, data management issues a RACROUTE REQUEST=DEFINE, CLASS=DATASET, DSTYPE=T macro.

RACF authorizes access to protected tape data sets through RACF authorization checking. RACF bypasses any tape data set password protection. If the tape data set is not RACF-protected or the tape protection option is not active, data management authorizes access to tape data sets by password protection.

Note: If you have a tape management system, you might find that the RACF TVTOC provides some of the same information that the tape management system provides. Therefore, to avoid duplication of information, you might consider running your system with TAPEVOL inactive and TAPEDSN active as described in the following section.

Tape Data Set Protection (TAPEDSN Active and TAPEVOL Inactive)

- Tape volumes have no RACF protection.
- Using the ADDSD command with the TAPE operand gives only a profile in the DATASET class; there is no tape volume profile or TVTOC. This means:
  - No checking of the RACF security retention period (the number of days that must elapse before the data set can be deleted or overwritten)
  - No RACF integrity for full 44-character data set names
  - No RACF protection for multiple data sets on a volume

  Many installations use their own tape library management systems, which often provide the preceding functions. When RACF tape data set protection is active and the TAPEVOL class is inactive, installations must depend on the tape management system for the preceding functions. In this situation, installations must carefully maintain the integrity and security of their tape management systems, because RACF tape data set security depends on the integrity and security of the tape management system.
- Data management calls RACF in the DATASET class.

If you do not have a tape management system, your tape data set protection has no integrity when TAPEVOL is inactive. Your data sets may appear to have protection, but RACF cannot verify that the full data set name is correct.

With TAPEDSN active and TAPEVOL inactive, data management still uses RACROUTE REQUEST=AUTH when tapes are opened for input, and RACROUTE REQUEST=DEFINE when tapes are opened for output. However, RACF does not use the TVTOC during its processing, but assumes that information such as data set name, file (data set) sequence number, and tape volume label are correct.

Tape Volume Protection (TAPEVOL Active and TAPEDSN Inactive)

A tape volume is RACF-protected when it is explicitly defined to RACF for protection. Tape volumes are defined to RACF by (1) an authorized user issuing the RDEFINE command without the TVTOC operand, or (2) RACROUTE REQUEST=DEFINE when the TAPEVOL class is active and the PROTECT parameter is specified on a JCL DD statement or during EOV processing.

RACF authorizes access to protected tape volumes through RACF authorization checking. RACF bypasses any tape data set password protection. If the tape volume is not RACF-protected, or the SETROPTS TAPEDSN option is not active, data management authorizes access to tape data sets by password protection.

If RACF authorizes a user to access an explicitly defined tape volume, the user has access to all of the tape data sets on the volume. Therefore, you should only place tape data sets that have similar RACF authorization requirements on the same volume.

- You can protect tapes only by volume, by using discrete tape volume profiles (PROTECT=YES or the RDEFINE command). However, you can specify PROTECT=YES for multiple data sets on the same volume (the profile is reused).
- Using the ADDSD command for a tape data set results in an error message.
- Data management calls RACF in the TAPEVOL class.

No Tape Volume or Tape Data Set Protection (TAPEVOL Inactive and TAPEDSN Inactive)

- You have no RACF protection for data on tape, either by volume or by data set.
- Using the ADDSD command for a tape data set results in an error message. However, you can use the RDEFINE, RALTER, RDELETE, and RLIST commands for tape volume profiles, which provide protection if TAPEVOL or TAPEDSN are activated.
- Data management does not call RACF.

Protecting Existing Data on Tape (SETROPTS TAPEDSN in Effect)

To protect an existing data set on a tape volume, issue the ADDSD command with the TAPE operand. (This requires that the TAPEDSN option be in effect.) If the data set is *cataloged,* you need to specify only the data set name.

The following example shows how to protect a cataloged tape data set named USER01.TEST.DATA:

```
ADDSD 'USER01.TEST.DATA' TAPE
```

If the cataloged tape data set resides on more than one volume (a multivolume tape data set), RACF uses the data set name specified on the ADDSD command and the information supplied in the catalog to protect the data set on all of the volumes on which it resides.

To protect an existing tape data set that is *uncataloged,* issue the ADDSD command with the TAPE operand and specify:

- The data set name
- The tape volume on which the data set resides
- The unit name
- The file sequence number of the data set on the tape

For example, suppose you want to protect an uncataloged tape data set named USER03.TEST.DATA with a discrete RACF profile. The data set resides on a tape volume labeled 123456 and has a file sequence of 1. To protect this data set, enter:

```
ADDSD 'USER03.TEST.DATA' TAPE UNIT(TAPE) VOLUME(123456) FILESEQ(1)
```

From this information, RACF builds a discrete profile for both the data set and the tape volume. When you issue the ADDSD command to protect an existing tape data set, RACF creates an automatic tape volume profile. For more information, see “Tape Volume Profiles That Contain a TVTOC” on page 182.

Note that when you issue the ADDSD command to RACF-protect an uncataloged tape data set, you protect that data set only on the volume that you specify.

Data Sets

If you have an uncataloged tape data set that resides on more than one volume, you can RACF-protect this data set with a discrete profile using several commands. For example, suppose you want to RACF-protect a tape data set named USER02.TEST.DATA that resides on volumes 111111 and 222222.

1. To protect that portion of the data set residing on volume 111111, issue the ADDSD command:

   ```
   ADDSD 'USER02.TEST.DATA' TAPE UNIT(TAPE) VOLUME(111111) FILESEQ(1)
   ```

2. To protect that portion of the data set residing on volume 222222, issue the ALTDSD command with the ADDVOL operand as follows:

   ```
   ALTDSD 'USER02.TEST.DATA' ADDVOL(222222)
   ```

   Notes:

   a. You can protect only one volume at a time with the ALTDSD command and the ADDVOL operand. If your data set resides on more than two volumes, issue the ALTDSD command and specify the appropriate volume on the ADDVOL operand for each additional volume. For a tape data set with an entry in the TVTOC, the maximum number of volumes the data set can span is 42.

   b. Before you can use the ALTDSD command to protect a portion of a multivolume data set, at least one other portion of that data set must already be RACF-protected.

   c. RACF ignores this command if you specify a generic profile name for the data set.

Protecting New Data on Tape

Your installation can provide RACF protection for new tape data sets by using one or more of the following methods.

- A user can specify PROTECT=YES on the JCL DD statement when creating a new tape data set.

  RACF builds a discrete profile for the newly created data set and the tape volume on which the data set resides (unless the tape volume is already defined with the TVTOC option).

  When TAPEDSN and TAPEVOL are both active, a discrete data set profile is defined. If the tape volume profile already exists, the TVTOC is updated. If the tape volume profile does not already exist, RACF defines an automatic tape volume profile with a TVTOC.

  When TAPEDSN is not active and TAPEVOL is active, a discrete TAPEVOL profile is defined.

- You can assign the ADSP attribute to a user and issue the SETROPTS ADSP command.

  When the user creates a new tape data set, RACF automatically builds a discrete profile for the data set as well as the tape volume on which the data set resides (unless the tape volume is already defined with the TVTOC option).

Protecting Tape Volumes

This section describes how to RACF-protect tape volumes using the RDEFINE command with or without the TVTOC operand.

You can provide RACF protection for tape data sets by creating tape volume profiles that protect the tape volumes on which the data sets reside.

Defining Tape Volumes With a TVTOC: To provide protection for tape data sets, you (or an assigned administrator) can predefine individual tape volumes to RACF

using the RDEFINE command with the TAPEVOL class and TVTOC operand. Tape volumes defined with the RDEFINE command and TVTOC operand are called *scratch pool volumes*.

When RACF processes the RDEFINE command with the TVTOC operand, it places the user ID of the command issuer in the access list of the volume with ALTER authority. A scratch pool volume can be used by any RACF-defined user for output (for writing). When the first user writes a data set to a scratch pool volume, RACF places the user ID of that user in the access list of the volume with ALTER authority. After RACF creates the volume's access list, only the command issuer, the first user of the volume, and any users added to the access list with UPDATE authority can write additional data sets to the volume.

For example, to define a tape volume labeled TX0050 with the attribute that it can hold a TVTOC and assign it a UACC of NONE, enter:

```
RDEFINE TAPEVOL TX0050 TVTOC UACC(NONE)
```

After you define a tape volume with a TVTOC, you can use generic profiles to protect data sets that reside on that volume. To define a generic profile for data sets, use the ADDSD command and specify the profile name.

The following example shows how to define the generic profile USER03.*.

```
ADDSD 'USER03.*'
```

Notes:

1. The user ID of the issuer of RDEFINE is placed automatically on the access list with ALTER only if SETROPTS ADDCREATOR is in effect.
2. The TAPEVOL class must be active for the RDEFINE command to succeed. For more information, see "Activating Tape Volume Protection (CLASSACT(TAPEVOL) Option)" on page 127.
3. The TVTOC operand applies only to discrete tape volume profiles.
4. When you issue the RDEFINE command with the TVTOC operand, you create a nonautomatic tape volume profile. For more information, see "Tape Volume Profiles That Contain a TVTOC" on page 182.
5. When you issue the ADDSD command, you can predefine a generic data set profile, or define a generic profile after the data set and TVTOC entry have been created. You can also use existing generic profiles that were created to protect DASD data sets. If you are using generic data set profiles for tape data sets, you should specify a retention period in those profiles because the SETROPTS retention period is not used.
6. The access authorities that apply to tape volume profiles are as follows:

   NONE Allows no access to data on the tape volume.

   READ Allows users to read from the tape volume.

   UPDATE Allows users to read from the tape volume, and to write additional data sets to the volume.

   CONTROL Is equivalent to UPDATE.

   ALTER Allows users to read from the tape volume, to write additional data sets to the volume, and to create or destroy tape volume labels through OPEN or end-of-volume operations. For discrete tape volume profiles, allows users to change the profile, including the access list.

Authorizing Access to a Data Set on a Tape Volume with a TVTOC: RACF maintains an entry in the TVTOC for each data set that a user writes to a scratch pool volume. The data set can be:

- Protected by a discrete profile, an appropriate generic profile, or both
- Not protected by any profile

When a user requests access to a data set on the tape volume, RACF performs access checking as follows:

1. RACF checks the user's authority to the volume on which the data set resides. If the user has sufficient authority to the volume, RACF grants access to the data set. If the user does not have sufficient authority to the volume, access checking proceeds with Step 2.
2. RACF checks to see if the data set is RACF-indicated. (For more information on RACF-indicated data sets, see "Protection Through Discrete Profiles" on page 155.) If the data set is RACF-indicated, access checking proceeds with Step 3.

   If the data set *is* RACF-indicated, RACF checks for a discrete profile that protects the data set. If RACF finds a discrete profile and the user has sufficient authority to the data set, RACF grants access. If the user does not have sufficient authority to the data set, RACF denies access. If RACF does not find a discrete profile, access checking proceeds with Step 3.
3. If the data set is RACF-indicated but RACF does not find a discrete profile, RACF searches for an appropriate generic profile. If RACF finds a generic profile, RACF grants or denies access based on the user's authority. If RACF does not find a generic profile, RACF fails the request.
4. If the data set is *not* RACF-indicated, RACF searches for an appropriate generic profile that protects the data set. If RACF finds a generic profile and the user has sufficient authority to access the data set, RACF grants the request. If the user does not have sufficient authority to access the data set, RACF fails the request.

   If RACF does not find a generic profile, the data set is not RACF-protected and, therefore, any user can access the data set.

Defining Tape Volumes Without a TVTOC: You can also define tape volumes without using the TVTOC operand. When you define a tape volume in this manner, RACF does not maintain a TVTOC to control access to data sets on the volume. Instead, RACF controls access to data sets on the tape volume using only the access list in the volume's profile. Users with at least READ authority to the volume can read any data on the tape. Users with at least UPDATE authority to the volume can write data on the tape.

The following sequence of commands shows how to define a tape volume without a TVTOC and how to control access to the data sets on that volume.

1. To define and protect a tape volume, issue the RDEFINE command with the appropriate operands and assign a UACC of NONE to the volume.

   ```
   RDEFINE TAPEVOL profile-name UACC(NONE)
   ```

   For example, to define a tape volume labeled 123456 and assign it a UACC of NONE, issue the following command:

   ```
   RDEFINE TAPEVOL 123456 UACC(NONE)
   ```

   The RDEFINE command adds a profile for the tape volume to the RACF database.

2. To allow a user access to the volume for the purpose of creating data sets, issue the PERMIT command with the appropriate operands and give the user UPDATE access authority. For tape volume 123456, enter the command as follows:

   ```
   PERMIT 123456 CLASS(TAPEVOL)
   ID(userid or groupname) ACCESS(UPDATE)
   ```

   UPDATE access authority allows a user to read and write data sets to the tape volume. You should *not* assign ALTER access authority to a general user because ALTER allows a user to overwrite the tape label.
3. If other users wish to access the data on the tape volume, issue the PERMIT command with the appropriate operands and access authority. For example, to give another user READ access authority to tape volume 123456, issue the following command:

   ```
   PERMIT 123456 CLASS(TAPEVOL) ID(userid or groupname) ACCESS(READ)
   ```

   Note that a user must have sufficient authority to issue the PERMIT command. Because you gave the user who requested the tape volume UPDATE access authority, that user does not have sufficient authority to allow other users to access the tape volume.
4. When a user has finished working with the tape volume, issue the PERMIT command and specify the RESET(ALL) operand. RESET(ALL) deletes the entire current standard and conditional access lists from the tape volume's profile. For tape volume 123456, enter the command as follows:

   ```
   PERMIT 123456 CLASS(TAPEVOL) RESET(ALL)
   ```

   If you delete only the access lists from a tape volume profile, you retain RACF protection for data on the volume. (In this case, no users can access the data.) If you delete the tape volume profile itself, you have no RACF protection for data on the volume. (Any user can access the data.)

Security Levels and Security Categories for Tapes

As an option, you can add security level and security category protection to the tape volume's profile. To achieve this additional protection, use the RALTER command with the appropriate operands.

For example, to add a security level of CONFIDENTIAL and security categories of PLANNING and STATUS to the profile of tape volume 123456, enter:

```
RALTER TAPEVOL 123456 SECLEVEL(CONFIDENTIAL)
       ADDCATEGORY(PLANNING, STATUS)
```

When a user creates data sets on tape volume 123456, the user should ensure that each data set has the same security level and security categories as specified on the RALTER command.

To delete the security level and all of the security categories from the volume's profile (for example, when a user has finished working with a tape volume), issue the RALTER command with the NOSECLEVEL and DELCATEGORY(*) operands. For tape volume 123456, enter the command as follows:

```
RALTER TAPEVOL 123456 NOSECLEVEL DELCATEGORY(*)
```

Security Labels for Tapes

To add a security label of LABEL1 to the profile of tape volume 123456, enter:

```
RALTER TAPEVOL 123456 SECLABEL(LABEL1)
```

Data Sets

When a user creates data sets on tape volume 123456, the user should ensure that each data set has the same security label (LABEL1) as was specified on the RALTER command that defined the tape volume.

Note: When the SECLABEL class is active, and both the SETROPTS MLS(FAILURES) and SETROPTS MLACTIVE options are in effect, the security label of the tape volume profile is used for all data sets on the volume. For tape volume profiles without TVTOCs, the security label is assigned when the tape volume profile is defined. Tape volume profiles with TVTOCs are assigned a security label when the first data set is written to the tape. To see which related products are required, see "B1 Configuration Requirements" on page 12.

To delete the security label from the volume's profile (for example, when a user has finished working with a tape volume), issue the RALTER command rather than the RDELETE command with the NOSECLABEL operand. For tape volume 123456, enter the command as follows:

```
RALTER TAPEVOL 123456 NOSECLABEL
```

Tape Volume Profiles That Contain a TVTOC

If tape data set protection and the TAPEVOL class are both active, RACF creates and maintains a tape volume table of contents (TVTOC) that is part of the tape volume profile in the RACF database. The following section describes the TVTOC.

Tape Volume Table of Contents (TVTOC)

RACF creates and maintains the TVTOC for tape volumes that:

- Are defined using the RDEFINE command with the TVTOC operand
- Contain tape data sets protected by using the ADDSD command
- Contain tape data sets protected by specifying PROTECT=YES on the JCL DD statement
- Contain tape data sets created by a RACF-defined user who has the ADSP attribute

The TVTOC contains the following information:

- The number of data sets on the volume
- The full 44-character name used when creating the data set (from the DSN operand of the JCL DD statement)
- The volume serial number of each volume on which the data set resides (from the VOL operand of the DD statement)
- The file (data set) sequence number (from the LABEL operand of the JCL DD statement)
- The RACF internal data set name (from the naming conventions table or an installation exit routine)
- The data set creation date
- For each data set on the volume, an indicator that is set if the data set is protected by a discrete profile

You can list the information in the TVTOC of a tape volume profile by using the RLIST command. For more information, see *z/OS SecureWay Security Server RACF Command Language Reference*.

RACF makes entries in the TVTOC when a user:

- Opens a new data set on a predefined tape volume, or

- RACF-protects a new or existing tape data set

RACF then uses the information during access checking.

Notes:

1. The maximum number of entries for data sets that a TVTOC can contain is 500.

   Attention
   Processing that creates large numbers of TVTOC entries and large access lists, for example, could result in an attempt to exceed the maximum profile size.

2. The maximum number of volumes that any data set on the tape with an entry in the TVTOC can span is 42.
3. The maximum number of volumes that any data set on tape without a TVTOC can span is limited only by the maximum profile size.

When both TAPEDSN and TAPEVOL are active, RACF can create two different types of TVTOC profiles:

- An automatic TVTOC tape volume profile
- A nonautomatic TVTOC tape volume profile
- The NOSET option on the DELDSD command can be used to remove a discrete tape data set profile without deleting the tape volume profile. For more information, see *z/OS SecureWay Security Server RACF Command Language Reference*.

The sections that follow describe these profiles.

Automatic TVTOC Tape Volume Profiles

RACF creates an *automatic TVTOC tape volume profile* when one of the following occurs:

- A RACF-defined user has the ADSP attribute and creates a tape data set on a non-RACF-defined tape volume.
- A RACF-defined user creates a tape data set on a non-RACF-defined tape volume by specifying PROTECT=YES on the JCL DD statement.
- A RACF-defined user protects an existing tape data set on a non-RACF-defined tape volume using the ADDSD command with the appropriate operands.

When RACF creates an automatic tape volume profile, RACF does not use modeling, except possibly for the owner field as specified below. The tape volume profile that RACF creates contains the following fields:

- Owner—The user ID creating the profile, unless a different owner is specified by REQUEST=DEFINE or an ADDSD command, or a discrete data set profile is being created and the model profile specifies an owner
- Universal access authority (UACC)
- Access list—The creating user ID with ALTER authority
- Audit criteria—FAILURES(READ)
- RESFLG—Indicates the profile is automatic
- TVTOC—The tape volume table of contents

You can change any of these fields by using the RALTER or PERMIT command. (The most likely change is adding other users to the access list so that they can define data sets on the tape volume.)

When the security retention periods for all data sets on a volume that is protected by an automatic tape volume profile have expired and a user uses the volume for output, RACF deletes the volume's profile. When a user creates a new data set on such a tape volume and specifies PROTECT=YES on the JCL DD statement or has the ADSP attribute, RACF creates a new discrete tape volume profile with a TVTOC and generates a discrete profile for the data set. If the user does not specify PROTECT=YES on the JCL DD statement or have the ADSP attribute, RACF does not create new profiles for the volume or the data set. Therefore, the volume and any data sets on it are no longer RACF-protected and any user can read or write data on the volume.

Nonautomatic Tape Volume Profiles

RACF creates a *nonautomatic tape volume profile* when:

- A user predefines a tape volume using the RDEFINE command with the TVTOC operand, or
- A user modifies an automatic tape volume profile with the ALTDSD or PERMIT command

When the security retention periods for all data sets on a volume that is protected by a nonautomatic tape volume profile expire, RACF does not delete the volume's profile. However, the volume that is protected by the profile can be reused. As users write new data sets to the volume, RACF updates the volume's TVTOC to reflect the addition of new data sets even if the data sets are not RACF-protected.

Predefining Tape Volume Profiles for Tape Data Sets

Rather than defining individual tape volumes for use by specific users, installations can predefine scratch pool volumes with tape volume profiles for use by any user. An installation tape librarian can predefine tape volume profiles to RACF by using the RDEFINE command with the TVTOC operand and optionally, the SINGLEDSN operand. The TVTOC operand indicates that RACF creates a TVTOC the first time the tape is opened for output. The SINGLEDSN operand indicates that the tape volume can contain only one data set.

Predefining tape volumes when TAPEVOL and TAPEDSN are both active has the following advantages:

- To get a tape volume profile with a TVTOC, users do not have to have ADSP, use PROTECT=YES in the JCL, or manually define tape data sets with the ADDSD command.
- It is easier for users to use generic profiles for tape data sets. (If a user creates a tape data set and the user has ADSP or specifies PROTECT=YES, RACF always creates a discrete profile for the tape data set.)

To predefine tape volumes, the installation tape librarian selects new or newly degaussed tape volumes in the scratch pool for use with RACF tape data set protection. The librarian defines these tape volumes to RACF with a nonautomatic discrete profile by using the RDEFINE command and the TVTOC operand. (If you do not specify the TVTOC operand, the default is NOTVTOC.) RACF puts the user ID of the person who defines the tape volume (presumably the tape librarian) in the access list with ALTER authority. This action gives the librarian complete control over the profile and the tape volume. RACF puts the user ID in the access list with ALTER authority only if SETROPTS ADDCREATOR is in effect. If SETROPTS

NOADDCREATOR is in effect, the tape librarian needs to ensure that they are the owner of the profile and should issue the PERMIT command to give themselves ALTER authority if they need to have complete control over the profile and the tape volume.

When the first user creates a tape data set on a predefined tape volume, RACF builds a TVTOC in the tape volume profile and places the user ID of this person in the access list with ALTER authority. If the volume is defined with the SINGLEDSN operand, no one can write additional data sets on the volume. If the volume is not defined with the single-data-set option, only this user (and the tape librarian) can add additional data sets to the volume without further authorization. Other users can add data sets to the volume only if they have been placed in the volume's access list with at least UPDATE authority.

When the tape librarian needs more tape volumes for the scratch pool, the librarian can issue the SEARCH command with the EXPIRES operand to find tape volumes for which the security retention period for all of the data sets is expired (or close to expiring). The librarian can then use the RDELETE and RDEFINE commands to redefine these tape volumes.

Unlike DASD data sets, tape data sets are not deleted. A tape data set exists until it is overwritten by another program or by a utility such as IEHINITT. Specifying DISP=DELETE for a tape data set only causes the data set to be uncataloged if it was cataloged. DISP=DELETE does not remove RACF protection from the data set or delete the data on the tape volume.

RACF Security Retention Period Processing (TAPEDSN Must Be Active)

Before RACF allows a user to write to a tape that is protected by a tape volume profile containing a TVTOC, RACF checks whether the security protection for the current data on the tape volume has expired. To determine whether the RACF security retention period has expired, RACF uses one of the following:

- The RACF security retention period stored in the data set profile (specified using the RETPD operand on the ADDSD or ALTDSD command)
- If the data set profile does not contain a security retention period, one of the following:
  - For discrete profiles, RACF uses the creation date stored in the TVTOC and the default security retention period established by your installation using the RETPD operand on the SETROPTS command.
  - For generic profiles, RACF uses a zero value. This results in the data set being expired. For generic profiles, the default security retention period is *not* checked. Therefore, you must ensure that all generic profiles that protect tape data sets include a retention period. (Make sure to specify the RETPD operand on the ADDSD command for generic profiles.)

If a user wishes to overwrite an existing tape data set with a data set having a different name before the existing data set's RACF security retention period has expired, the user must do one of the following:

- Explicitly delete the data set profile using the DELDSD or RDELETE command
- Have at least UPDATE authority to the volume

If the user has sufficient authority to a tape volume or tape data set, the user can overwrite an existing data set using one of the following:

- The same data set name

- A data set name defined to RACF to which the user has authority
- A data set name not defined to RACF

If the RACF security retention period for an existing tape data set has not expired and the user does not have sufficient authority to overwrite it, RACF issues a message indicating that the user does not have sufficient authority to the volume or data set.

When a user specifies PROTECT=YES on the JCL DD statement, RACF updates the TVTOC to reflect the creation of the new data set. RACF also generates a discrete profile to protect the new data set and deletes any existing discrete profile that protected the overwritten data set.

A user can specify the security retention period for a tape data set by one of the following methods:

- For a data set protected by either a discrete or generic profile, by using the RETPD operand on the ADDSD or ALTDSD command
- For a data set protected by a discrete profile, by specifying the EXPDT or RETPD operand on the JCL DD statement

For discrete profiles, if a user does *not* specify a security retention period for a tape data set, the retention period can be provided by one of the following:

- Profile modeling
- An installation exit routine
- A system-wide default set by the RETPD operand on the SETROPTS command

For generic profiles that protect tape data sets, the user must assign a security retention period to the profile by specifying the RETPD operand on the ADDSD or ALTDSD command. (If the security retention period is omitted, a zero value is used and the profile is treated as if it expired.)

When RACF is installed, the default security retention period is RETPD(0). If your installation specifies a different default security retention period for tape data sets, RACF uses the specified value in any of the following situations:

- When a user specifies RETPD=0 on the JCL DD statement
- When a user specifies EXPDT=current-date on the JCL DD statement
- When a user does not specify the EXPDT/RETPD JCL operands

Note: The RACF security retention period is independent of the data set retention period specified by the EXPDT/RETPD JCL operand. However, the two retention periods are initially the same if the user who creates the data set has ADSP or specifies PROTECT=YES on the JCL DD statement. You can modify the security retention period in the data set profile by using the ALTDSD command.

If a tape volume contains more than one data set, RACF protects each data set independently. RACF achieves this protection by not allowing users with UPDATE authority to one or more of the data sets to rewrite any data set until one of the following occurs:

- The profiles for all of the data sets that sequentially follow that data set on the tape volume have been deleted.
- The security retention periods for all of the data sets that sequentially follow that data set on the tape volume have expired.

Note, however, that users who have at least UPDATE authority to the volume can write to the volume unconditionally.

In response to RDELETE or DELDSD commands, RACF deletes tape volume profiles and the discrete tape data set profiles for all data sets residing on tapes when all of the data sets that the TVTOC points to have expired. For generation data groups (GDGs), RACF does not automatically delete RACF protection of the volumes containing the oldest generation when a new generation is defined. Because residual data remains on a tape volume even after the security retention period of the RACF profiles has expired, installations should consider degaussing tape volumes on which all of the data sets have an expired security retention period. The librarian can then redefine these tape volumes to RACF using the RDEFINE command with the TVTOC operand and, thereby, reenter the volumes into the common scratch pool.

Authorization Requirements for Tape Data Sets When Both TAPEVOL and TAPEDSN Are Active

When TAPEVOL is active, users with ALTER authority to a tape volume have full control over the volume profile, including the volume's access list. ALTER authority gives the user the ability to create and delete data sets on the volume and rewrite the tape volume label.

To open a RACF-protected tape data set for input (for reading), the user must have at least READ authority to the data set or the volume. When a RACF-protected volume is opened for input and the user does not have the authority necessary to write to the data set, a message may be issued to the system operator to remove the write-enable ring (file protect ring). (The authority necessary to open a RACF-protected tape data set for output is described below.) For more information, see "IEC.TAPERING Profile in the FACILITY Class" on page 188.

To open a RACF-protected tape data set for output (for writing), the user must have UPDATE authority to the tape volume, or the following authority:

- **To rewrite or add to a data set without changing the data set name,** the user requires UPDATE authority to the data set. If the data set is not the last data set on the tape volume, all of the subsequent data sets must have passed their security retention periods or be explicitly deleted using the DELDSD or RDELETE command.
- **To overwrite an existing data set on a tape with a data set of a different name,** the security retention periods for the data set and any subsequent data sets must have expired. The user must also have authority to create a data set with the specified name (the authority checks are the same as for DASD data sets).
- **To add a new data set to the end of a tape,** the user requires UPDATE authority to the tape volume, and the volume profile must allow more than a single data set. The user must also have authority to create a data set with the specified name (the authority checks are the same as for DASD data sets).

Note: If a data set is in the TVTOC of a tape volume profile, but is not covered by a discrete profile, a generic profile, or an entry in the global access checking table, the data is not RACF-protected.

Authorization Requirements for Tape Data Sets When TAPEVOL Is Inactive and TAPEDSN Is Active

To open a RACF-protected tape data set for input (for reading), the user must have at least READ authority to the data set. When a RACF-protected tape data set is opened for input and the user does not have the authority necessary to write to the data set, a message may be issued to the system operator to remove the write-enable ring (file protect ring). (The authority necessary to open a RACF-protected tape data set for output is described below.) For more information, see “IEC.TAPERING Profile in the FACILITY Class”.

To open an existing RACF-protected tape data set for output (for writing), the user must have at least UPDATE authority to the data set. To create a new tape data set for output or, to catalog an existing tape data set after opening it for output, the user must have ALTER authority to the data set.

Authorization Requirements for Tape Data Sets When TAPEVOL Is Active and TAPEDSN Is Inactive

When TAPEVOL is active, users with ALTER authority to a tape volume have full control over the volume profile, including the volume’s access list. ALTER authority gives the user the ability to create and delete data sets on the volume and to rewrite the tape volume label.

To open a tape data set on a RACF-protected tape volume for input (for reading), the user must have at least READ authority to the tape volume. When a data set on a RACF-protected tape volume is opened for input and the user does not have the authority necessary to write to the data set, a message may be issued to the system operator to remove the write-enable ring (file protect ring). (The authority necessary to open a tape data set on a RACF-protected volume for output is described below.) For more information, see “IEC.TAPERING Profile in the FACILITY Class”.

To open a tape data set on a RACF-protected tape volume for output (for writing), the user must have at least UPDATE authority to the volume.

JCL Changes

To protect tape data sets, installations should provide generic profiles or code PROTECT=YES (when TAPEVOL is active) for each data set that requires protection. Data management allows you to specify PROTECT=YES for each data set on a tape volume.

Installations with HSM

The hierarchical storage manager (HSM) works with the volume level of protection, so HSM tape volume profiles should not contain a TVTOC.

Because a TAPEVOL profile can span a maximum of 10,000 tape volumes, HSM includes a way to extend its pool of backup and migration tape volumes. To take advantage of this extension method, add the DFHSM profile to the HSMHSM profile in the TAPEVOL class.

IEC.TAPERING Profile in the FACILITY Class

Depending on the release of DFSMS/MVS, the type of tape drive, and any tape management system that are installed on your system, you can allow users to open tape data sets for input without removing the write-enable ring (or equivalent) by

creating a profile to protect a resource called IEC.TAPERING in the FACILITY class and allowing users to have READ access authority to this resource.

Attention

You should only allow access to the IEC.TAPERING resource for users who can be trusted not to abuse the authority to write to tapes they are allowed to read.

For IBM 3490 tape drives and for IBM 3480 tape drives with the IDRC feature, this profile is not checked. Instead, the tape device cannot use WRITE operations when the user has only READ authority.

See the following example for setting up an IEC.TAPERING profile:

1. Create a profile to protect the IEC.TAPERING resource:

   ```
   RDEFINE FACILITY IEC.TAPERING UACC(NONE)
   ```

   Note: If you wish to allow this for all users on your system, specify UACC(READ) and omit the following PERMIT command.

2. Permit users or groups, as appropriate:

   ```
   PERMIT IEC.TAPERING CLASS(FACILITY) ID(userid or groupname)
          ACCESS(READ)
   ```

For more information, see *z/OS DFSMS: Using Magnetic Tapes*.

Password-Protected Tape Data Sets

Your installation can provide password protection to data sets that reside on tape volumes. When you define a tape volume to RACF using the RDEFINE command, data management does not verify password protection when a user requests access to a data set residing on this tape volume.

When you define a tape volume to RACF through ADSP or PROTECT=YES in the JCL statement, the user or operator must supply the correct password for the password-protected tape data set on the volume.

Passwords should be maintained for tape data sets on RACF-protected tape volumes if they are to be used (1) on systems that do not have RACF installed or (2) on RACF systems where tape protection might not be active.

To maintain passwords for tape data sets, password information can be specified on the LABEL operand along with the PROTECT parameter on the same JCL DD statement. Also, the password indicators in tape header and trailer labels are set for RACF-protected tape volumes based on password information specified on the LABEL operand. All password-protected data sets on a RACF-protected tape volume must have the same level of password protection.

Using the PROTECT Parameter for Tape Data Set or Tape Volume Protection

To define a tape data set or a tape volume to RACF for protection by a discrete profile, specify the PROTECT parameter on the JCL DD statement that identifies a tape data set on the volume. If generic profile checking is active, do not use the

PROTECT parameter unless a discrete profile is required for the data set. If the data set is also password-protected, the password must be supplied before the tape volume is RACF-protected.

Multivolume Tape Data Sets

When a multivolume tape data set is opened for input, RACF performs authorization checking for those volumes that are RACF-protected.

When a multivolume tape data set is opened for output, and the first volume is RACF-protected, all succeeding volumes are RACF-protected as part of the same volume set.

When attempting to effect changes to multivolume tape data sets, use the RALTER command rather than the RDELETE command. If the resource you specify is a member of a tape volume set, RACF deletes the definitions for *all* of the volumes in the set.

A single profile is maintained in the RACF database for a tape volume set. Thus all volumes in the volume set share the same access list and the same statistics and auditing options. (For additional information on protecting multivolume tape data sets, see “Protecting Existing Data on Tape (SETROPTS TAPEDSN in Effect)” on page 177.)

RACF Authorization of Bypass Label Processing (BLP)

At system initialization and JES initialization, an installation can specify whether the system supports bypass label processing (BLP).

If your system does not support BLP processing, the system converts all BLP requests to requests for nonlabeled tapes. If a labeled tape is mounted to satisfy this specification, RACF performs authorization checking and, if the user has sufficient authority, the label is destroyed. For more information, see “Tape Data Set and Tape Volume Protection for Nonlabeled (NL) Tapes” on page 191.

If your system supports BLP processing, RACF provides installations with the ability to control the use of the BLP option on JCL DD statements. To control who can use BLP, take the following steps:

1. Activate the TAPEVOL class.
2. Define a profile in the FACILITY class to protect the ICHBLP resource, and grant users READ or UPDATE authority, as appropriate.

   To open a tape for input and bypass label processing when the TAPEVOL class is active, the user must have at least READ authority to the volume (if the volume is defined) as well as to the ICHBLP resource in the FACILITY class.

   To open a tape for output and bypass label processing, the user must have at least UPDATE authority to the volume (if the volume is defined) as well as to the ICHBLP resource in the FACILITY class.

RACF checks the user’s authority to the ICHBLP resource when the user attempts to access a tape with an IBM standard or ANSI label (even if BLP is specified on the LABEL operand of the DD statement for the tape volume).

RACF performs BLP authorization checking only if the TAPEVOL class is active. If TAPEVOL is not active, data management does not call RACF to perform BLP or tape access checking.

If RACF finds an ICHBLP profile, RACF verifies that the user has sufficient authority to use bypass label processing. If the user does not have sufficient authority, RACF fails the request.

If RACF does not find an ICHBLP profile or if the user has sufficient authority to use bypass label processing, RACF performs authorization checking on the volume. If the user has sufficient authority to the volume, RACF grants the request. Otherwise, RACF fails the request.

Note: RACF performs authorization checking on a volume based on the volume serial number specified on the JCL statement. Proper authorization checking, therefore, depends on the operator mounting the correct volume.

Authorization Requirements for Labels

The following rules apply to RACF-protected tape volume labels:

- To rewrite a tape volume label without additional authority, the user must have ALTER authority to the volume.
- To destroy a tape volume label (when converting from standard labels to nonstandard or nonlabeled tapes), the user must have ALTER authority to the volume.
- To create a tape volume label (when writing a standard labeled data set to a nonlabeled or nonstandard labeled volume), the user must have ALTER authority to the volume.

Tape Data Set and Tape Volume Protection with Nonstandard Labels (NSL)

Data management does not do authorization checking for nonstandard labeled tapes. However, if the user is going to destroy standard labels, data management calls RACF to determine whether the tape volume is protected. If the tape is RACF-protected and TAPEDSN is active, the user must have ALTER authority to the volume and to the data sets on the volume. For more detailed information, consult the DFSMS/MVS publications.

Tape Data Set and Tape Volume Protection for Nonlabeled (NL) Tapes

Opening an Unlabeled Tape for Input

To open an unlabeled tape for input, the user must have at least READ authority to one of the following:

- To the volume
- If TAPEDSN is active, to the data sets that may have previously been defined to RACF within the TVTOC for the volume. Because this is a nonlabeled tape volume, RACF does not automatically create or maintain TVTOC data set entries. However, TVTOC data set entries can be manually created and maintained with the ADDSD command.

If data management finds a labeled tape when opening the volume for input, it rejects the request.

Opening an Unlabeled Tape for Output

To open an unlabeled tape for output when TAPEDSN is active, the user must have at least UPDATE authority to the tape volume and to all data sets on the volume. If a labeled tape is encountered when opening an output tape and the user has

ALTER authority to the volume, the volume label is destroyed and RACF protection for the volume is deleted. For additional details, see *z/OS DFSMS: Using Magnetic Tapes*.

You should be careful when using nonspecific volume requests for output volumes because the operating system assigns volume serial numbers and it is impossible for you to determine if the volume mounted is defined to RACF under a different number. If your installation plans to use RACF protection for nonlabeled tapes, you should use JCL scans to prevent the use of nonspecific volume requests and establish procedures to ensure that operators mount the correct tapes.

Notes:

1. Because protection is on a volume level, you should specify PROTECT=YES in the DD statement for only one data set on the volume.
2. RACF protection of nonlabeled tapes does not depend on tape data set protection being active.
3. RACF does not maintain the TVTOC for nonlabeled tapes. But if the TVTOC contains any entries, RACF uses these entries during authorization checking.
4. You cannot RACF-protect nonlabeled tapes that have a volume serial number of "L*nnnnn*".

Chapter 7. Protecting General Resources

This chapter provides in-depth information on protecting general resources.

RACF-protected resources can be divided into two categories: data sets and general resources. General resources are all of the resources that are defined in

the class descriptor table. For example, general resources include DASD and tape volumes, load modules (programs), terminals, and others.

Defining Profiles for General Resources

The RACF commands that you can use to work with general resource profiles are shown in Table 14.

Table 14. RACF Commands Used to Work with General Resource Profiles

Activity	Command
Defining	RDEFINE
Listing	RLIST
Changing	RALTER
Allowing or denying access	PERMIT with CLASS(*classname*)
Searching	SEARCH with CLASS(*classname*)
Deleting	RDELETE
Note: For the authority needed to issue any of these commands, see *z/OS SecureWay Security Server RACF Command Language Reference*.	

Summary of Steps for Defining General Resource Profiles

This summary presents the steps required by RACF to define general resource profiles. Please note that specific instructions for most resources supported by the supplied general resource classes are contained elsewhere in this book.

1. Determine which resources are to be protected by the profile. This involves the following information:
   - The general resource class, such as TAPEVOL or TERMINAL
   - The profile name:
     – If you specify a generic profile name, the profile can protect more than one resource.

       Using generic profiles instead of discrete profiles can greatly reduce the effort of maintaining the profiles. In general, you should create generic profiles to cover the majority of resources, using discrete profiles only for exceptions.

       Also, you should consider creating a profile to be used as a model, especially if you are specifying complex access lists. Models can be used when creating any kind of resource profile (discrete or generic), and modeling can be done across classes. To model, specify the FROM operand on the RDEFINE command. To model across classes, you should also specify the FCLASS operand. Before using modeling, see “Possible Changes to Copied Profiles When Modeling Occurs” on page 40.

       Note: To specify generic profile names, either generic command processing or generic profile checking (the SETROPTS GENCMD or SETROPTS GENERIC option) must be in effect for the class. For example, for the TERMINAL class:

       ```
       SETROPTS GENERIC(TERMINAL)
       ```

     – If you specify a discrete profile name, the profile can protect only one resource.
     – The rules for specifying profile names for most supplied general resource classes are described elsewhere in this book.

Notes:

a. For some kinds of resources, (such as terminals, DASD volumes, and CICS, IMS, and SDSF classes), you should consider using resource group profiles instead of generic profiles. Creating resource group profiles can save a significant amount of work. See “Creating Resource Group Profiles” on page 221 for more information.

b. You can use RACFVARS profiles to specify values for variables (indicated by an ampersand &) in profile names. For more information, see “Using RACF Variables in Profile Names (RACFVARS Class)” on page 226.

- Decide which access is to be allowed to all users on the system who are not otherwise limited. In RACF, this is called the universal access authority (UACC). This has the same meaning as the access authority on access lists (see step 3 on page 198). In most cases, the UACC should be NONE or READ.
- Decide which user or group is to be the owner of the new resource profile. By default, this is the user who creates the profile.
  - If the owner is a user, the owner can list, modify, or delete the resource profile. Note that being the owner of a resource profile does not, by itself, allow a user to have access to the resource or resources that are protected by the profile. For more information, see step 16 on page 636 in “Authorizing Access to RACF-Protected Resources” on page 634.
  - If the owner is a group, the authority of a user who has a group-level attribute in that group (such as group-SPECIAL or group-AUDITOR) extends to resources that are protected by this profile.
- Decide which user, if any, should be notified by a message when users make unsuccessful attempts to access resources that are protected by the profile (NOTIFY operand).
- Decide whether RACF should log access attempts to resources that are protected by the profile (AUDIT operand).

  Note: To see the results of the logging done by RACF, use the RACF report writer. For more information, see *z/OS SecureWay Security Server RACF Auditor's Guide*.
- If your installation is using some form of security classification, do one of the following:
  - If security labels are used on your system, decide which security label (if any) to assign to the profile.

    When security labels are being used on your system, be aware that security levels and categories are ignored.
  - If security levels are used on your system, decide which security level (if any) to assign to the profile.
  - If security categories are used on your system, decide which security categories (if any) to assign to the profile.
  - If your installation has written RACF installation exits to use the LEVEL operand, decide which value to specify for LEVEL.
- Depending on the class of the resource, the profile might have additional fields for which you should assign values. For example:
  - Profiles in the APPCLU class have SESSION segments
  - Profiles in the TAPEVOL class have the SINGLEDSN and TVTOC operands

– Profiles in the TERMINAL and GTERMINL classes have the WHEN and TIMEZONE operands (both optional). WHEN determines the times and days a terminal may be used.

Note: This WHEN is not the same as the WHEN operand in a conditional access list.

See the appropriate section of this book or the description of the RDEFINE command in *z/OS SecureWay Security Server RACF Command Language Reference* for specific information on these operands.

- To copy an existing profile, specify the name of the existing profile on the FROM operand. If the existing profile is in a different class, specify FCLASS also.

2. Create the general resource profile using the RDEFINE command:

   ```
   RDEFINE classname profile-name other-operands
   ```

   Note: To change a general resource profile, use the RALTER command.

3. If specific users or groups are to have specific access to the resource, use the PERMIT command to create one or both of the access lists:
   - Each entry in the standard access list states which access (such as NONE or READ) a specific user or group has:

     ```
     PERMIT profile-name CLASS(classname)
            ID(userid or group) ACCESS(access-authority)
     ```

   - Each entry in the conditional access list states which access (such as NONE or READ) a specific user or group has, and also states which condition a user must meet to get the specified access:

     ```
     PERMIT profile-name CLASS(classname)
            ID(userid or group) ACCESS(access-authority)
            WHEN(condition)
     ```

   Notes:

   a. Access authorities that you can specify with UACC or specifically assign to users vary from class to class, and are described in the sections of this book that describe the specific classes.

   b. Not all classes are described in this book (for example, the DSNR class is not described in this book). Also, in some classes (notably the FACILITY class), the access required by some resource managers to specific profiles is described in the documentation of the resource manager. Therefore, go to that resource manager to find the descriptions of that class. In the case of DSNR, which is used by DATABASE 2, see the description of the class descriptor table (CDT) in *z/OS SecureWay Security Server RACF Macros and Interfaces* and the DB2 books.

   c. The profile creator may be automatically added to the access list with ALTER authority. For more information, see “Automatic Addition of Creator’s User ID to Access List” on page 142.

4. If you have not already done so, activate the resource class:

   ```
   SETROPTS CLASSACT(classname)
   ```

5. For performance benefits, consider doing one of the following:
   - Allow all users on the system to have access to the resource at some level (such as READ or UPDATE) by creating a global access checking table entry that has a name similar to the new resource profile.

     See “Setting Up the Global Access Checking Table” on page 206.

- Reduce I/O to the RACF database by requesting that RACF keep all profiles in the class in storage:

  ```
  SETROPTS RACLIST(classname)
  ```

  Note: This is required for some classes.

Choosing Between Discrete and Generic Profiles in General Resource Classes

Most general resource classes (except the PROGRAM class) give you a choice of creating either a discrete profile or a generic profile. Refer to “Protecting Load Modules as Controlled Programs” on page 250 for more information on creating profiles in the PROGRAM class.

Choose a *generic profile* to protect more than one resource with the same security requirements.

Notes:

1. You can use the characters *, **, or % to specify in which way (if at all) the resources protected by the profile have identical characters in their names.
2. If you use the character & in a profile name, there must be a corresponding RACFVARS profile. See “Using RACF Variables in Profile Names (RACFVARS Class)” on page 226.

Choose a *discrete profile* to protect one resource with unique security requirements. The name of a discrete profile has no generic characters.

Choosing Among Generic Profiles, Resource Group Profiles, and RACFVARS Profiles

Table 15 gives some considerations for choosing among generic profiles, resource group profiles, and RACFVARS profiles.

Table 15. Choosing Among Generic Profiles, Resource Group Profiles, and RACFVARS Profiles

How to Choose	Reference
Use generic profiles when the names of the resources have logically matching characters.	“Rules for Generic Profile Names” on page 200 and *z/OS SecureWay Security Server RACF Command Language Reference*.
Use resource group profiles if the names of the resources do not have logically matching characters and there is a resource grouping class (such as GTERMINL or GDASDVOL).	“Creating Resource Group Profiles” on page 221.
Use RACFVARS profiles if the names of the resources do not have logically matching characters and there is no resource grouping class.	“Using RACF Variables in Profile Names (RACFVARS Class)” on page 226.

Using Generic Profiles

In each address space, RACF keeps up to four lists of generic profiles that have been referenced. Each list comprises one DATASET high-level qualifier, or one general resource class based on the value of KEYQUAL in the class descriptor table (CDT) entry for that class (assuming the class is not RACLISTed in some way).

When RACF needs to reference a set of generic profiles that are not present in the address space, the oldest list is deleted and the new list replaces it. The performance impact of doing this can be especially important during the OPEN for a concatenated DD statement. If possible, group data sets with the same high-level qualifier together in the concatenation, so that RACF does not need to read the same list of generics multiple times. Also, consider using global access checking for commonly referenced data sets, because RACF does not need to use the generic profiles if the access is granted by global access checking.

When RACF loads the list of generic profile names, significant I/O to the RACF database may occur. Therefore, the number of generic profiles within a data set high-level qualifier or general resource class should be kept as small as practical, which may suggest the use of discrete profiles instead of generics. The performance of generics in RACF is optimized for the case where each generic profile protects several (possibly many) resources for the average case.

When searching a list of generic profiles to find a match, RACF scans the list in sequential order, which can cause high CPU usage if the list is a long one. This can be especially important when using RACLISTed profiles in CICS or IMS. With RACLISTed profiles, discrete entries are located quickly via a binary tree search, but the generics must be scanned sequentially. This effect can be exacerbated if the generic profile names (or member names, for RACLISTed classes) begin with RACF variable names, for example &VAR.ABC*.

Rules for Generic Profile Names

There are a few rules that apply to naming generic profiles.

When You Can Specify Generic Profile Names

You can create a profile with a generic name when either of the following is true for the class of the profile:

- The SETROPTS GENERIC option is in effect. Not only does this option allow the creation of generic profiles, it also causes RACF to use generic profiles during authorization checking.
- The SETROPTS GENCMD option is in effect. In this case, generic profiles can be created and modified, but RACF does not use them during authorization checking. This is intended for use when migrating from discrete profiles to generic profiles.

Generic Naming

Some of the generic profile naming for general resources has been enhanced with some of the same concepts as generics for data set profiles. You may now have an asterisk (*) within a profile name, representing one qualifier of a resource name. You may also use a double asterisk (**) to represent zero or more qualifiers within a general resource generic profile or at the end of such a profile. Use of the double asterisk (**) in general resource generic profiles is not controlled by the SETROPTS EGN option which applies only to the data set profiles.

Some of the rules for generic characters are different between general resource and data set generic profiles. See *z/OS SecureWay Security Server RACF Command Language Reference* for more information.

Other Rules for Generic Profile Names

The following rules apply to profile names:

- Valid generic characters are *, %, and **:

- Specify % in the profile name to match any single non-blank character (except a period) in the same position of the resource name.
- Specify * or ** in the profile name to match more than one character in the same position of the resource name. For a complete description and examples of how to specify * and **, see *z/OS SecureWay Security Server RACF Command Language Reference*.

Limited Use of %* in General Resource Profile Names

The %* combination requires special attention.

New profiles with an ending %* are not allowed nor are profiles named %*. The RDEFINE command returns an error message.

Existing profiles with an ending %* are usable, but they should be deleted before creating any new profiles with a middle or beginning * or **. The RALTER and RDELETE commands accept %* to enable you to make the changes.

Instead of using an ending %*, create new profiles ending with * for similar function (change `AB.C%*` to `AB.C*`).

If you have an existing profile whose entire name is %*, you should create a new profile whose new name is **.

Notes:

1. The above considerations also apply to generic members of grouping classes.
2. When creating the new profiles, consider using the FROM operand for continued use of the same access list.

- For any particular general resource class, the profile naming conventions are defined by how the resource name is specified on the call to RACF. When your application programmers are designing the resource names for use in their invocations of the security product, they should be aware of the problems involved with using *, %, or & in resource names. For more information, see *z/OS SecureWay Security Server External Security Interface (RACROUTE) Macro Reference*.

  As you define general resource profiles, users must observe the naming conventions for that particular class. For some classes, the naming conventions are described in this book. However, other products (both IBM and non-IBM) can issue RACROUTE REQUEST=AUTH. You must check the documentation produced for those products for authoritative information on how those products call RACF. You should gather the following information from the calling product's documentation:

  - When the call to RACF is done. In other words, what user action causes the call to RACF?

    Some further questions to ask: Also, are there settings in the product that cause the call to occur? Are there installation exits that can prevent the call, or change the results of the call?
  - What is the class name used on the call to RACF?
  - What is the resource name used on the call to RACF? If you are using discrete profiles, this is the profile name. If you are using generic profiles, you

need to know how many qualifiers (portions of the name that are separated by periods) there are, and what the qualifiers mean, so that you can specify meaningful profile names.

Note: If you do not follow the resource naming convention established by the caller of RACF, you could create profiles that are never used. For example, if you create a discrete profile with less than the correct number of qualifiers, the profile is never used during RACF authorization checking.

– What do the access authorities (READ, UPDATE, CONTROL, ALTER) mean? Remember, these values are hierarchical (UPDATE is higher than READ, and so forth), and do not necessarily mean what the English word means. For example, for terminals, READ means “allowed to logon”, not “allowed to read information”.

Generic Profile Checking of General Resources

The rules for access-authorization checking of generic profiles for general resources are similar to those for the DATASET class.

- Generic profiles are not checked unless generic profile checking is in effect for the class. To do this, issue the following command:

  `SETROPTS GENERIC(`*`classname`*`)`
- If the class is not active, RACF does not check for profiles. RACF returns the default return code of the class to the resource manager. For a complete description, see “Authorization Checking for RACF-Protected Resources” on page 633 .
- If more than one profile covers a particular resource, RACF searches for profiles in the following order:
  – Discrete profile
  – Matching generic profiles (see Table 16)

Table 16. Sample General Resource Profile Names in Order from Most Specific to Least Specific

		Resources Being Accessed			
Profile Name	**Profile Type**	**MEDIUM**	**MEDIUM.PAPER**	**MEDIUM.PAPER.TEST**	**MEDIUM.ONLINE.FINAL**
MEDIUM.A	Discrete				
MEDIUM.ONLINE.FINAL	Discrete				X
MEDIUM.ONLINE.*	Generic				X
MEDIUM.PAPER	Discrete		X		
MEDIUM.PAPER.TEST	Discrete			X	
MEDIUM.PAPER.%	Generic				
MEDIUM.PAPER.*	Generic			X	
MEDIUM.PAPER.**	Generic		X	X	
MEDIUM.PAPER%	Generic				
MEDIUM.PAPER*	Generic		X	X	
MEDIUM.PAPE%	Generic		X		
MEDIUM.PAP*	Generic		X	X	
MEDIUM.PRINT.*	Generic				
MEDIUM.&X (where: &X = PAPER in RACFVARS profile)	Generic		X		

Table 16. Sample General Resource Profile Names in Order from Most Specific to Least Specific (continued)

Profile Name	Profile Type	Resources Being Accessed			
		MEDIUM	MEDIUM.PAPER	MEDIUM.PAPER.TEST	MEDIUM.ONLINE.FINAL
MEDIUM.&Y (where: &Y = ONLINE.FINAL in RACFVARS profile)	Generic				X
MEDIUM.%APER	Generic		X		
MEDIUM.*.FINAL	Generic				X
MEDIUM.*.FINAL*	Generic				X
MEDIUM.**.FINAL	Generic				X
MEDIUM.**.PAPER	Generic		X		
MEDIUM.*	Generic		X	X	X
MEDIUM.**	Generic	X	X	X	X
MEDIUM*.**	Generic	X	X	X	X
.	Generic		X	X	X
*.**	Generic	X	X	X	X
*	Generic	X	X	X	X
**	Generic	X	X	X	X

To determine which profiles have the potential to protect any particular resource, use the FILTER or MASK operands on the SEARCH command to generate a list of profiles that might match the resource. For example, you might specify the user's user ID on the FILTER operand to limit the list of profiles displayed. Some examples follow:

```
SEARCH CLASS(TAPEVOL)  FILTER(**.userid.**)
SEARCH CLASS(JESSPOOL) FILTER(**.userid.**)
```

In general, the list of profiles generated by the SEARCH command is the order in which RACF searches for a matching profile. To review the list:

1. Find all profiles that match the resource name.
2. If no profile names match, check for profile names that include an ampersand (&) (RACF variables). You must list the RACFVARS profile to determine the value of a RACF variable:

   ```
   RLIST RACFVARS variable-name
   ```

   Also, the SEARCH command does not list grouping profiles (such as GTERMINL) that protect the resource. To do this, use the RESGROUP operand on the RLIST command.

   ```
   RLIST member-class resource-name RESGROUP
   ```

   See "Which Profiles Protect a Particular Resource?" on page 223.

   If these methods do not find a profile, the resource is not protected.
3. If only one profile matches, it protects the resource.
4. Otherwise, find two profiles that both match the resource name. Then, compare them character by character. Where they first differ, if one has a discrete character and the other has a generic character, the one with the discrete character wins. If both have a generic character where they differ and:
   - If one has an & and the other has a %, *, or **, the & wins.

- If one has a % and the other has a * or **, the one with % wins.
- If one has a * and the other has a **, the one with * wins.

Note: The following is generally true:

Given two generic profiles that match a resource, the one whose first generic character is farther from the beginning of the name is used.

If two profile names match except for one character position, the following is the order in which RACF examines them:

```
blank
.
$  (X'5B')
#  (X'7B')
@  (X'7C')
A through Z
0 through 9
&  (X'50')
%
*
```

For example, the following profile names all match in the first three character positions (A.B), and are shown in the order RACF examines them:

```
A.B
A.B.B
A.BA
A.BZ
A.B0
A.B9
A.B&X
A.B%
A.B*
```

When in doubt about the search order, create sample profiles and check the order of profile names shown by the SEARCH command.

Granting Access Authorities

You can grant (or deny) user or group access to a RACF-protected resource either explicitly, by assigning the specific user or group access authority with the appropriate command, or implicitly, with the universal access authority (UACC).

Each resource that you protect with RACF requires a UACC, which is the default access authority for the resource. All users in the system who are not specifically authorized in the access list of that resource profile, except users defined with the RESTRICTED attribute, can still access the resource with the authority specified by UACC (unless the UACC is NONE). These users include users not defined to RACF.

Note: Users with the RESTRICTED attribute can access the resource when they are specifically authorized in the access list with the sufficient authority.

If you specifically assign a user or group an access authority to a resource, the specified authority overrides the UACC specified for the resource.

Valid authorities that you can specify with UACC or specifically assign to users or groups vary from class to class, and are described in the sections of this book that describe the specific classes.

Note: Not all classes are described in this book (for example, the DSNR class is not described in this book). Also, in some classes, the access required by some resource managers to specific profiles is described in the documentation of the resource manager.

Table 17 shows additional meanings for several access authorities for general resources.

Table 17. ALTER, NONE, and CONTROL, UPDATE, and READ Access Authorities for General Resources

ALTER	For discrete profiles, the specified user or group has full control over the resource and the resource profile, and can authorize other users and groups to access the resource. For generic profiles, only the profile owner, users with the SPECIAL attribute, and group-SPECIAL users whose groups own the profile have control over the resource profile and can authorize other users and groups to access the resource. For both profiles, full resource access is allowed.
NONE	The specified user or group is not permitted to access the resource or list the profile.
CONTROL, UPDATE, READ	These access authorities allow listing of selected portions of the profile and grant resource access in a variety of ways, depending on the class.

Limiting the Size of Your Access Lists

If you need to authorize a large number of users to a resource, you must consider the limitations on the size of the access list. The access list of each profile is limited to 65535 bytes. Each user or group you add to the access list uses 11 bytes. Therefore, the maximum number of entries is 5957. To minimize the impact of these limitations, you can create groups and add the groups, rather than the individual users, to the access list.

There are additional considerations if you need to authorize a large number of users for a resource in a class that can be processed to an in-storage profile using the SETROPTS RACLIST command or the RACROUTE REQUEST=LIST macro. A single in-storage profile is limited to 65535 bytes. Each entry in the access list uses 9 bytes in storage. Therefore, the maximum number of access list entries is 7273, a larger number than the same profile can contain on the database. However, because the in-storage profile includes other information in addition to the access list, such as installation data, application data, and the conditional access list, the maximum number of entries in the access list may be fewer than 7273.

If you use resource member and grouping profiles, you should define a given member name only once. If you define the same member name more than once, for example, in multiple grouping profiles using the ADDMEM command or in both a member profile and a grouping profile, it will be difficult to determine the resulting security attributes for that member after RACLIST processing merges the profiles. RACF also merges the access lists of each profile, making it difficult for you to determine the number of access-list entries you have used. In addition, the combined number of access-list entries may cause the profile to become too large to be processed, and RACLIST processing may fail.

Conditional Access Lists for General Resource Profiles

You can require that a user or a job have entered the system from a particular device when accessing general resources. Specifically:

- You can require that a user be logged onto a particular terminal by specifying WHEN(TERMINAL(...)) on the PERMIT command.

  The TERMINAL class must be active for this support to take effect.
- You can require that a user be logged onto a particular console by specifying WHEN(CONSOLE(...)) on the PERMIT command.

  The CONSOLE class must be active for this support to take effect.
- You can require the batch job accessing the resource to have been submitted from a particular JES input device by specifying WHEN(JESINPUT(...)) on the PERMIT command.

  The JESINPUT class must be active for this support to take effect.
- You can require that a user enter the system from a particular partner LU by specifying WHEN(APPCPORT(...)) on the PERMIT command.

  The APPCPORT class must be active for this support to take effect.

Note: If an access list contains more than one condition, *any* of the conditions allows the specified access. For example, if you enter the PERMIT command with WHEN(CONSOLE(01) TERMINAL(20)) specified, you allow the access when *either* console 01 *or* terminal 20 is used.

Example:

To ensure that an operator (or group of operators) can issue certain operator commands only when logged on at a particular console, enter:

```
PERMIT profile-name CLASS(OPERCMDS) ID(user or group) ACCESS(READ)
       WHEN(CONSOLE(console-id))
```

You can require a user to access a program from a particular system by specifying WHEN(SYSID(*system-identifier*)) on the PERMIT command:

```
PERMIT profile-name CLASS(PROGRAM) ID(user or group) ACCESS(READ)
       WHEN(SYSID(system-identifier))
```

This conditional access list entry is only valid for the PROGRAM class.

See “Program Control by System Identifier (SMFID)” on page 252 for more information.

Setting Up the Global Access Checking Table

You can use global access checking to improve the performance of RACF authorization checking for selected resources. Global access checking should be used for *public resources* that are accessed frequently. For example, an entry in the global access checking table can allow all users on the system to have READ access to the SYS1.HELP data set.

The global access checking table is maintained in storage and is checked early in the RACF authorization checking sequence. If an entry in the global access checking table allows the requested access to a resource, RACF performs no further authorization checking. This can avoid I/O to the RACF database to retrieve a resource profile, and can result in substantial performance improvements.

Global access checking is used for authorization processing invoked by the RACROUTE REQUEST=AUTH macro. It is not used for authorization processing invoked by the RACROUTE REQUEST=FASTAUTH macro.

How Global Access Checking Works

When a user requests access to a resource for which a RACROUTE REQUEST=AUTH macro is issued, and global access checking is in effect for the class of the resource, RACF searches the global access checking table for a matching entry. If there is a matching entry, RACF compares the access authority requested by the user (READ, UPDATE, CONTROL, or ALTER) to the access authority associated with the resource in the global access checking table.

If the requested access is less than or equal to the authority specified in the table entry for the resource, global access checking grants the requested access immediately, without checking the profile protecting the resource. Otherwise, normal RACF authorization checking is performed. Global access checking can only permit accesses, not deny them.

Global access checking is bypassed for users who have the RESTRICTED attribute. See “Defining Restricted User IDs” on page 87 for more information.

Attention

Because RACF performs global access checking before many of the other kinds of access authority checks, such as security label checking or access list checking, global access checking might allow access to a resource you are otherwise protecting. To avoid a security exposure to a sensitive resource, do not create an entry in the global access checking table for a resource that is protected by a profile containing a security level, security category, or security label. (If the security label in the profile is SYSLOW, a global access checking table entry with an access authority of READ can be created.)

Candidates for Global Access Checking

The following resources are candidates for *public* access and therefore for global access checking:

- SYS1.BRODCAST
- SYS1.HELP
- SYS1.PROCLIB
- ISPF/PDF libraries
- ISPF libraries (panels, skeletons, tutorial, and so forth)
- Tools libraries

Note: User IDs with the RESTRICTED attribute that require access to resources like these must be specifically authorized in the access list for each required resource with sufficient access authority.

Creating Global Access Checking Table Entries

To create an entry in the global access checking table:

1. Plan the entries for the global access checking table, using the following guidelines:
   a. Identify resource profiles that are accessed frequently and for which a performance benefit is desired.

b. If there are resource profiles with UACC other than NONE, consider adding similar entries to the global access checking table. Using this “matched pair” approach, each entry would have the same name as a profile, and the access specified in the entry would generally match the UACC of the profile. Do *not* add a global access checking table entry if any of the following are true:

- The profile has a security level, security category, or security label (other than SYSLOW).

  Note: If the profile has a security label of SYSLOW, the global access checking table entry can have an access of READ.
- The profile has an entry in the standard access list that is *lower* than the access level of the global access checking table entry.
- The profile has an entry in a conditional access list that is *more restrictive* than the access level of the global access checking table entry.
- The profile requests auditing of successful access attempts at or below the level specified in the corresponding global access checking table entry.

For example, if you have a data set profile ‘PHONE.DIRECT’ with UACC(READ) and AUDIT(FAILURES(UPDATE)) specified, you might create a global access checking table entry for it as follows:

```
RALTER GLOBAL DATASET ADDMEM('PHONE.DIRECT'/READ)
```

However, if there are users or groups in the standard access list of profile ‘PHONE.DIRECT’ with an access authority of NONE (which is lower than the UACC), do *not* create a global access checking table entry. A global access checking table entry would allow these users and groups to read the phone directory.

c. If you have resources that are protected by a generic profile with UACC other than NONE, and others that are protected by a more specific (generic or discrete) profile that has specific access requirements such as an access list, consider adding two entries: one for the larger set of resources (with access authority equal to the UACC of the profile) and the other for the smaller set of resources (with access authority of NONE).

For example, if you have a profile of SYS1.** with a UACC(READ), but you also have some specific profiles with more restrictive entries, such as SYS1.XYZ with UACC(READ) and an access list with JOE/NONE, create two entries:

```
SYS1.XYZ/NONE
SYS1.**/READ
```

The entry with /NONE does not fail any attempts but stops requests for SYS1.XYZ from being granted by the SYS1.** entry.

See the examples later in this section for other possible entries.

2. Add the resource class to the global access checking table using the RDEFINE command with the GLOBAL operand and the class name:

   ```
   RDEFINE GLOBAL classname
   ```

3. To allow global access checking for a specific resource, add an entry to the global access checking table using the RALTER command as follows:

   ```
   RALTER GLOBAL classname ADDMEM(resource-name/access-level)
   ```

where:

resource-name
: is the equivalent of a profile name in the class specified. If generic command processing is in effect for *classname* (through the SETROPTS GENCMD command), *resource-name* on the ADDMEM operand can include the generic characters *, **, or %. In general, the rules for specifying these characters are the same as the rules for specifying these characters in generic profile names except that generic characters are allowed in any qualifier (even if not allowed in certain qualifiers of the profile names).

  After they have been added, generic entries in the global access checking table are used in global access checking even if generic profile checking is turned off (through the SETROPTS NOGENERIC command).

  The resource name can also include the name qualifiers `&RACUID` (RACF user ID) or `&RACGPID` (*current connect group*). For example, the following entry allows users to have ALTER access to data sets that begin with their own user IDs.

  The resource name can also include variables defined in the RACFVARS class. Note that when defining a resource name for a profile in a mixed-case class, you must enter the character strings `&RACUID`, `&RACGPID`, and any RACFVARS variable names in uppercase. For more information on RACFVARS usage, see "Using RACFVARS with Mixed-Case Classes" on page 230.

  ```
  RALTER GLOBAL DATASET ADDMEM('&RACUID.**'/ALTER)
  ```

  Note: The preceding entry does not *change* a user's access to his or her own data sets, but speeds the process by which RACF grants the access. (It also prevents any auditing of such access attempts.)

  The word `&RACGPID` allows the user's current connect group to be used in the same way. For example, the following allows all users to have READ access to group data sets for their current connect group:

  ```
  RALTER GLOBAL DATASET ADDMEM('&RACGPID.**'/READ)
  ```

  Note: If the current connect group is found in the global access table and list-of-groups processing is in effect, list-of-groups checking is ignored.

access-level
: can be NONE, READ, UPDATE, CONTROL, or ALTER.

For more information on specifying the ADDMEM operand, see *z/OS SecureWay Security Server RACF Command Language Reference*.

4. When you are finished updating the global access checking table, issue the SETROPTS command with the GLOBAL operand for each class affected:

   ```
   SETROPTS GLOBAL(classname)
   ```

Attention

Save a listing of the global access checking table. This can help you recover from the accidental deletion or alteration of the global access checking table or its entries. You can use the RLIST command to make this listing quickly.

IBM recommends that you write an EXEC that contains the commands you use to create the global access checking table. The EXEC should include the RLIST command to provide an independent record of the actual table created. Also, if the global access checking table is accidentally deleted (using the RDELETE command), the EXEC can readily be used to regenerate the table.

5. IBM strongly recommends for all classes that, for each entry in the global access checking table, you create a similar resource profile. Such a “matched pair” approach can help ensure the continuation of protection if global access checking becomes disabled. For example,

   ```
   RDEFINE classname resource-name UACC(access-level)
   ```

   At the end-of-volume (EOV) processing, RACROUTE REQUEST=AUTH is issued with OLDVOL specified for authority checking with the DATASET and TAPEVOL classes. This check bypasses the global access table checking and uses resource profile definitions for authority checking. By not having a “matched pair” approach, you may get different results.

Adding an Entry to the Global Access Checking Table

To add an entry to the global access checking table, issue the RALTER command with the ADDMEM operand, then refresh the in-storage global access checking table. For example:

```
RALTER   GLOBAL classname  ADDMEM(resource-name/level)
SETROPTS GLOBAL(classname) REFRESH
```

Deleting an Entry from the Global Access Checking Table

To delete an entry from the global access checking table, issue the RALTER command with the DELMEM operand, then refresh the in-storage global access checking table. For example:

```
RALTER   GLOBAL classname  DELMEM(resource-name/level)
SETROPTS GLOBAL(classname) REFRESH
```

Attention

Do not use the RDELETE command unless you intend to delete the entire global access checking table for that class.

Examples of Creating Global Access Checking Table Entries

The following examples show you how to create entries in the global access checking table for:

- The SYS1.HELP data set
- The SYS1.BRODCAST data set
- Group data sets
- The master catalog and user catalogs

Example 1: The SYS1.HELP Data Set: To allow all users to have READ access to SYS1.HELP, enter:

```
SETROPTS GLOBAL(DATASET)
RDEFINE  GLOBAL DATASET
RALTER   GLOBAL DATASET ADDMEM('SYS1.HELP'/READ)
ADDSD    'SYS1.HELP' UACC(READ)
SETROPTS GLOBAL(DATASET) REFRESH
```

Example 2: The SYS1.BRODCAST Data Set: To allow all users to have UPDATE access to SYS1.BRODCAST, enter:

```
SETROPTS GLOBAL(DATASET)
RDEFINE  GLOBAL DATASET
RALTER   GLOBAL DATASET ADDMEM('SYS1.BRODCAST'/UPDATE)
ADDSD    'SYS1.BRODCAST' UACC(UPDATE)
SETROPTS GLOBAL(DATASET) REFRESH
```

Example 3: Group Data Sets: To specify that all users are to have UPDATE access authority to data sets whose high-level qualifier is the user's current connect group, enter:

```
RDEFINE GLOBAL DATASET ADDMEM('&RACGPID.**'/UPDATE)
```

Note: &RACGPID entries provide more flexibility than the GRPACC attribute. Table 18 shows some points of comparison.

*Table 18. Comparison of GRPACC Attribute with &RACGPID.** Entry in Global Access Checking Table*

GRPACC Attribute	&RACGPID.** Entry
Applies only to group data set profiles created by the user while the user has the GRPACC attribute.	Applies to all group data sets for all users.
Always allows UPDATE access.	Can allow READ, UPDATE, CONTROL, or ALTER access.
Works by changing access lists in group data set profiles. These can be changed individually later.	Works the same way for all users in all connect groups. Changes affect all users.
Applies only to resources in the DATASET class.	Can apply to any class that might include a group name in profile names (such as TAPEVOL).

Example 4: The Master Catalog and User Catalogs: With the exception of a very select group, users should only be allowed to READ the master catalog. To allow this, enter:

```
RALTER GLOBAL DATASET ADDMEM('CATALOG.MASTER.**'/READ)
ADDSD  'CATALOG.MASTER.**' UACC(READ)
PERMIT 'CATALOG.MASTER.**' ID(SYSGROUP) ACCESS(CONTROL)
```

Notes:

1. The exact form of the names specified on these commands depends on the naming conventions at your installation. This example assumes that catalog names take the form:

   CATALOG.MASTER.MVSESA.V*vvvvvv* for a master catalog
   and
   CATALOG.*applic*.V*vvvvvv* for application-specific catalogs
   (for example, TSO or DB2)

2. The access authority that is required to update and maintain the catalog depends on the DFP release that is installed on your system.

General Resources

For user catalogs, most users should be allowed to add entries as they create data sets. To allow this, enter:

```
RALTER GLOBAL DATASET ADDMEM('CATALOG.**'/UPDATE)
ADDSD 'CATALOG.**' UACC(UPDATE)
```

Stopping Global Access Checking for a Specific Class

To stop global access checking for a specific class, issue:

```
SETROPTS NOGLOBAL(classname)
```

Listing the Global Access Checking Table

To list the global access checking table, do one of the following:

- For a list showing the entries in a particular class, enter the following command:

  ```
  RLIST GLOBAL classname
  ```

  This shows the entries in the order in which they are searched by RACF.
- See the DSMON report that lists the global access checking table described in *z/OS SecureWay Security Server RACF Auditor's Guide* for a list that shows all entries in the table.

Special Considerations for Global Access Checking

When using global access checking, consider the following:

- Global access checking is used for authorization processing invoked by the RACROUTE REQUEST=AUTH macro. It is not used for authorization processing invoked by the RACROUTE REQUEST=FASTAUTH macro.
- Global access checking is bypassed for access requests by users with the RESTRICTED attribute. See "Defining Restricted User IDs" on page 87.
- RACF authorization checking via RACROUTE REQUEST=AUTH searches the global access checking table for a matching entry, ignoring profiles in the class. If no global access checking table entry matches the search, or if the access specified in the entry is less than the access being requested, RACF then searches for a matching profile in the class. This processing occurs regardless of whether or not the class is RACLISTed (by either SETROPTS RACLIST or RACROUTE REQUEST=LIST).
- RACF searches the global access checking table for an entry that best matches the name of the resource, much as RACF searches for a matching profile. The output from the RLIST command shows the order used.
- The group resource classes (such as GTERMINL) are ineligible for global access checking.
- When global access checking allows a request to access a data set, that data set is considered to be protected by RACF, and therefore any OS password processing and prompting that would otherwise have occurred is bypassed.
- When global access checking allows a request, RACF maintains no statistics.
- When global access checking allows a request, RACF performs no logging other than that requested by the SETROPTS LOGOPTIONS command.
- RACF bypasses global access checking if the PROFILE, CSA, or PRIVATE operand is specified on the request for RACF authorization checking (RACROUTE REQUEST=AUTH).
- Updated global access checking table entries become effective with the next IPL or after execution of the SETROPTS command with the GLOBAL(*classname*) operand (with or without the REFRESH operand).

- The only use for an access of NONE in the global access table is to force RACF to look for a profile. This would typically be used when you have access list entries which have a lower access level than a data set's UACC, or when you want to ensure that auditing or security classification checking takes place for a specific data set.
- When RACF is enabled for sysplex communication, the SETROPTS GLOBAL and SETROPTS GLOBAL(*classname*) REFRESH commands are propagated to the other members of the sysplex data sharing group.

Field-Level Access Checking

You can use RACF to control which users can access data in RACF profiles at the field level through *field-level access checking*. To do this, you create profiles in the FIELD class and permit users to the profiles.

Using field-level access checking, you can:

- Allow a user or group to modify a particular field (or segment) in all profiles of a particular type. For example, you can define a profile to control access to the ACCTNUM field of the TSO segment of user profiles. If you give a user UPDATE authority to this profile, the user can modify the ACCTNUM field in all user profiles.
- Allow all users to read or modify a particular field (or segment) of their *own* user profiles. To do this, specify ID(&RACUID) on the PERMIT command.

Note: RACF command processors and panels support field-level access checking only for fields in segments other than the base segments of RACF profiles. However, the ICHEINTY and RACROUTE REQUEST=EXTRACT macros can support field-level access checking for fields in any segment of any RACF profile. If your installation has written its own programs that use these macros to access the RACF database, you can modify these programs to implement field-level access checking.

To use field-level access checking, take the following steps:

1. Define profiles in the FIELD class:

   ```
   RDEFINE FIELD profile-name UACC(NONE)
   ```

   where *profile-name* has the following format:

   ```
   profiletype.segmentname.fieldname
   ```

   where:

   profiletype
   is one of the following:
   - Class name for general resource profiles
   - DATASET for data set profiles
   - GROUP for group profiles
   - USER for user profiles

   segmentname
   is one of the following:
   - BASE for BASE segments (this is supported only by user-written code)
   - CICS for CICS segments
   - DCE for DCE segments

- DFP for DFP segments
- DLFDATA for DLFDATA segments
- LANGUAGE for LANGUAGE segments
- LNOTES for LNOTES segments
- NDS for NDS segments
- NETVIEW for NETVIEW segments
- OMVS for OMVS segments
- OPERPARM for OPERPARM segments
- OVM for OVM segments
- SESSION for SESSION segments
- SSIGNON for SSIGNON segments
- STDATA for STDATA segments
- SVFMR for SystemView segments
- TME for TME segments
- TSO for TSO segments
- WORKATTR for WORKATTR segments

Note: This is also the operand used on RACF commands to work with the segment.

fieldname
is the name of the field to be protected as described in Table 19 on page 215.

For example, to control access to *all* fields in the TSO segment of all user profiles, issue the RDEFINE command and specify USER.TSO.* as the profile name. Before issuing this command, however, check the Special Note below:

```
RDEFINE FIELD USER.TSO.* UACC(NONE)
```

When you specify a UACC of NONE, you prevent all users from accessing the TSO segment in all user profiles, including their own. Likewise, if you specify a UACC of READ, you allow all users to read the information contained in all fields of the TSO segment for all user profiles.

Special Note

Note that the profile name USER.TSO.* is a generic profile name. Before you issue the above command, generic profile checking for the FIELD class must be active. If it is not active, issue the SETROPTS GENERIC(FIELD) command before you define the generic profile.

To control access to specific fields in the TSO segment of user profiles, issue the RDEFINE command and specify the specific field as the third qualifier in the profile name. Use Table 19 on page 215 to determine which qualifier to use. For example, when changing the account number field in a TSO segment, users specify the ACCTNUM operand on the TSO option of the ALTUSER command:

```
ALTUSER userid TSO(ACCTNUM(account-number))
```

According to Table 19 on page 215, to control access to the ACCTNUM field, create a profile using the TACCNT qualifier:

```
RDEFINE FIELD USER.TSO.TACCNT UACC(NONE)
```

2. Allow specific users or groups to have the appropriate access to the field:

   ```
   PERMIT USER.TSO.TLPROC CLASS(FIELD) ID(TSOADM) ACCESS(UPDATE)
   ```

   The previous example shows how to create a profile that gives user ID TSOADM the authority to change the logon procedure (TLPROC field) in the profiles of all TSO users.

   Note: You can also specify the value `&RACUID` with the ID operand on the PERMIT command for FIELD profiles. When you enter this value on the PERMIT command, you allow all users access to the specified field or segment of their own user profiles. For example, if you issue the following command, you allow all users to read the TLPROC field in the TSO segment of their own user profiles.

   ```
   PERMIT USER.TSO.TLPROC CLASS(FIELD) ID(&RACUID) ACCESS(READ)
   ```

3. When you are ready to start using the protection defined in the profiles, activate the FIELD class:

   ```
   SETROPTS CLASSACT(FIELD)
   ```

   Note: If you do not activate the FIELD class and activate SETROPTS RACLIST processing for the FIELD class, only SPECIAL users can access fields in segments (other than the base segment) of RACF profiles.

4. You *must* activate SETROPTS RACLIST processing for the FIELD general resource class. For a complete description of this function, see “SETROPTS RACLIST Processing” on page 131.

   ```
   SETROPTS RACLIST(FIELD)
   ```

   Note: Once you activate SETROPTS RACLIST processing for the FIELD class, any time you make a change to a FIELD profile, you must also refresh SETROPTS RACLIST processing for the FIELD class for the change to take effect.

   ```
   SETROPTS RACLIST(FIELD) REFRESH
   ```

Table 19. Relationship of RACF Command Operands to FIELD Profile Names

To control the use of this operand: [1]	Use this qualifier in FIELD profiles:
CICS Segment in User Profiles:	
OPCLASS	OPCLASS and OPCLASSN [2]
OPIDENT	OPIDENT
OPPRTY	OPPRTY
TIMEOUT	TIMEOUT
XRFSOFF	XRFSOFF
DCE Segment in User Profiles:	
AUTOLOGIN	DCEFLAGS
DCENAME	DCENAME
HOMECELL	HOMECELL
HOMEUUID	HOMEUUID
UUID	UUID
DFP Segment in Data Set Profiles:	
RESOWNER	RESOWNER
DFP Segments in User and Group Profiles:	
DATAAPPL	DATAAPPL
DATACLAS	DATACLAS
MGMTCLAS	MGMTCLAS
STORCLAS	STORCLAS

Table 19. Relationship of RACF Command Operands to FIELD Profile Names (continued)

To control the use of this operand: [1]	Use this qualifier in FIELD profiles:
DLFDATA Segment in General Resource Profiles (DLFCLASS Class):	
RETAIN	RETAIN
JOBNAMES	JOBNAMES and JOBNMCNT [2]
LANGUAGE Segment in User Profiles:	
PRIMARY	USERNL1
SECONDARY	USERNL2
LNOTES Segment in User Profiles:	
LNOTES	SNAME
NDS Segment in User Profiles:	
NDS	UNAME
NETVIEW Segment in User Profiles:	
IC	IC
CONSNAME	CONSNAME
CTL	CTL
MSGRECVR	MSGRECVR
OPCLASS	OPCLASS and OPCLASSN [2]
DOMAINS	DOMAINS and DOMAINSN [2]
NGMFADMN	NGMFADMN
NGMFVSPN	NGMFVSPN
OMVS Segment in Group Profiles:	
GID	GID
OMVS Segment in User Profiles:	
ASSIZEMAX	ASSIZE
CPUTIMEMAX	CPUTIME
FILEPROCMAX	FILEPROC
HOME	HOME
MMAPAREAMAX	MMAPAREA
PROCUSERMAX	PROCUSER
PROGRAM	PROGRAM
THREADSMAX	THREADS
UID	UID
OPERPARM Segment in User Profiles:	
ALTGRP	OPERALTG
AUTH	OPERAUTH
AUTO	OPERAUTO
CMDSYS	OPERCMDS
DOM	OPERDOM
KEY	OPERKEY
LEVEL	OPERLEVL
LOGCMDRESP	OPERLOGC
MFORM	OPERMFRM
MIGID	OPERMGID
MONITOR	OPERMON
MSCOPE	OPERMSCP and OPERMCNT [2]
ROUTCODE	OPERROUT
STORAGE	OPERSTOR
UD	OPERUD
OVM Segment in Group Profiles:	
GID	GID

Table 19. Relationship of RACF Command Operands to FIELD Profile Names (continued)

To control the use of this operand: [1]	Use this qualifier in FIELD profiles:
OVM Segment in User Profiles:	
FSROOT HOME PROGRAM UID	FSROOT HOME PROGRAM UID
SESSION Segment in General Resource Profiles (APPCLU Class):	
CONVSEC INTERVAL LOCK SESSKEY	CONVSEC KEYINTVL SLSFLAGS SESSKEY
SSIGNON Segment in General Resource Profiles (PTKTDATA Class):	
KEYENCRYPTED KEYMASKED	SSKEY SSKEY
STDATA Segment in General Resource Profiles (STARTED Class):	
USER GROUP PRIVILEGED TRACE TRUSTED	STUSER STGROUP FLAGPRIV FLAGTRAC FLAGTRUS
SVFMR Segment in General Resource Profiles:	
PARMNAME SCRIPTNAME	PARMN SCRIPTN
TME Segment in Group Profiles:	
ROLES	ROLES and ROLEN [2]
TME Segment in Data Set Profiles:	
ROLES	ROLES and ROLEN [2]
TME Segment in General Resource Profiles	
ROLES	ROLES and ROLEN [2]
GROUPS	GROUPS and GROUPN [2]
RESOURCE	RESOURCE and RESN [2]
CHILDREN	CHILDREN and CHILDN [2]
PARENT	PARENT
TSO Segment in User Profiles:	
ACCTNUM COMMAND DEST HOLDCLASS JOBCLASS PROC MAXSIZE MSGCLASS SECLABEL SIZE SYSOUTCLASS UNIT USERDATA	TACCNT TCOMMAND TDEST THCLASS TJCLASS TLPROC TMSIZE TMCLASS TSOSLABL TLSIZE TSCLASS TUNIT TUDATA
WORKATTR Segment in User Profiles:	

Table 19. Relationship of RACF Command Operands to FIELD Profile Names (continued)

To control the use of this operand: [1]	Use this qualifier in FIELD profiles:
WANAME	WANAME
WABLDG	WABLDG
WADEPT	WADEPT
WAROOM	WAROOM
WAADDR1 through WAADDR4	WAADDR1 through WAADDR4
WAACCNT	WAACCNT

Notes:

1. Many operands in this table have corresponding versions that include a prefix of NO. In addition, several operands have corresponding versions that include prefixes of ADD and DEL. Refer to the *z/OS SecureWay Security Server RACF Command Language Reference* to identify these.
2. For full field-level access checking and authority protection, define all qualifiers.

Planning for Profiles in the FACILITY Class

The FACILITY class can be used for a wide variety of purposes depending on the products installed on your system. If the FACILITY class is active, users might need access to particular resources to perform specific tasks. Therefore, they must have access based on the profiles protecting those resources. For example:

- READ access to IEAVECTOR allows users to use the vector facility.
- READ access to ICHBLP allows tape users to bypass label processing.
- READ access to IEC.TAPERING allows tape users to write to tape data sets without removing the write-enable ring.
- There are many other resources that can be protected using RACF profiles in the FACILITY class for use with many different subsystems and products.

You should activate the FACILITY class for the first such profile that is required on your system. You can create FACILITY profiles as needed to control who can use a number of processes on your system.

IBM also recommends that you activate SETROPTS RACLIST processing for the FACILITY general resource class. When you activate this function, you improve performance because I/O to the RACF database is reduced. For a complete description of this function, see “SETROPTS RACLIST Processing” on page 131.

```
SETROPTS RACLIST(FACILITY)
```

Note: If you activate SETROPTS RACLIST processing for the FACILITY class, any time you make a change to a FACILITY profile, you must also refresh SETROPTS RACLIST processing for the FACILITY class for the change to take effect.

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

Delegating Authority to Profiles in the FACILITY Class

You can use several methods to allow another user, such as a tape librarian or storage administrator, to work with profiles in the FACILITY class:

- Assign the user as OWNER of all of the FACILITY profiles used by the function.
- Create a group representing the function and give the user group-SPECIAL authority within the group. Then assign the group as OWNER of the FACILITY profiles used by the group.

- If the SETROPTS GENERICOWNER option is in effect, give the user CLAUTH(FACILITY), create a “top” generic profile to which the user is assigned as OWNER. The SETROPTS GENERICOWNER option limits this user to creating FACILITY profiles that are more specific than the “top” generic profile.

  Note: It is recommended that you do *not* create a top profile of ** in the FACILITY class, as this could lead to problems with RJE.

  For more information about the GENERICOWNER option, see “Restricting the Creation of General Resource Profiles (GENERICOWNER Option)” on page 117.

For other examples for delegating authority in the FACILITY class, check the specific areas in this manual as shown in Table 20.

Table 20. Delegating Authority in the FACILITY Class

Situation	Reference
To allow users to obtain dumps when they are using programs to which they only have EXECUTE authority, using the IEAABD.DMPAUTH resource	“Protecting Program Dumps Using the FACILITY Class” on page 267
To allow users to open tape data sets for input without removing the write-enable ring (or equivalent), using the IEC.TAPERING resource	“IEC.TAPERING Profile in the FACILITY Class” on page 188
To allow users to access DFP-controlled DASD or tape data sets when those data sets are neither cataloged nor system temporary data sets, using ICHUNCAT.*dataset-name* and CATDSNS	“Preventing Access to Uncataloged Data Sets (CATDSNS Option)” on page 124
To allow migration of security functions from JES into RACF, using the RJE, RJP, and NJE NODES profiles	“Understanding NODES Profiles” on page 453

Providing the Ability to List User Information

You can allow a general user to use the LISTUSER command to list information in the base segment of a USER profile. To provide this ability, define a profile to protect the IRR.LISTUSER resource in the FACILITY class. It is recommended that you define this profile with UACC(NONE) until you determine which users you would like to give the ability to list user information. If the profile does not exist, standard LISTUSER privileges will apply when RACF determines whether the command issuer is authorized.

Attention

Make sure an existing generic profile in the FACILITY class does not inadvertently grant this authority by default! The safest approach is to protect the IRR.LISTUSER resource with UACC(NONE) until you determine which users should be granted authority to list user information.

A general user can list the base segment of another user’s USER profile if the command issuer has READ access to the IRR.LISTUSER resource in the FACILITY class. However, this authority does not apply when the target of the LISTUSER command has any of the SPECIAL, AUDITOR, or OPERATIONS attributes.

RACF does not log failed access attempts to IRR.LISTUSER. Rather, these attempts are logged as LISTUSER command violations. Successful accesses to IRR.LISTUSER are logged at the installation's discretion.

Providing the Ability to Reset User Passwords

You can allow a general user to use the ALTUSER command to resume user IDs and reset user passwords. To provide this ability, define a profile to protect the IRR.PASSWORD.RESET resource in the FACILITY class. It is recommended that you define this profile with UACC(NONE) until you determine which users you would like to give the ability to resume user IDs and reset user passwords. If the profile does not exist, standard ALTUSER privileges will apply when RACF determines whether the command issuer is authorized.

Attention

Make sure an existing generic profile in the FACILITY class does not inadvertently grant this authority by default! The safest approach is to protect the IRR.PASSWORD.RESET resource with UACC(NONE) until you determine which users should be granted authority to resume user IDs and reset user passwords.

Users with READ access to the IRR.PASSWORD.RESET resource in the FACILITY class can:

- Use the PASSWORD operand, provided the user being altered does not have the SPECIAL, OPERATIONS, or AUDITOR attribute.
- Use the RESUME operand, without specifying a date, provided the user being altered does not have the SPECIAL, OPERATIONS, or AUDITOR attribute.

Users who have UPDATE access to the IRR.PASSWORD.RESET resource in the FACILITY class can use the NOEXPIRED operand (in conjunction with the PASSWORD operand), provided the user being altered does not have the SPECIAL, OPERATIONS, or AUDITOR attribute. Being the owner of the USER profile does not provide sufficient authority to use the NOEXPIRED operand.

Users who have the SPECIAL attribute can use the NOEXPIRED operand (in conjunction with the PASSWORD operand) for all users, including users who have the SPECIAL, OPERATIONS, or AUDITOR attribute. Having the group-SPECIAL attribute does not provide sufficient authority to use the NOEXPIRED operand.

RACF does not log failed access attempts to IRR.PASSWORD.RESET. Rather, these attempts are logged as ALTUSER command violations. Successful accesses to IRR.PASSWORD.RESET are logged at the installation's discretion.

Example 1

A group of help desk administrators requires the ability to view user profile information, reset users' passwords, and resume user IDs that have been revoked. These administrators are members of the HELPDESK group in RACF. The following commands provide the necessary authorization to the HELPDESK group:

```
RDEFINE  FACILITY IRR.LISTUSER UACC(NONE)
PERMIT   IRR.LISTUSER CLASS(FACILITY) ID(HELPDESK) ACCESS(READ)

RDEFINE  FACILITY IRR.PASSWORD.RESET UACC(NONE)
PERMIT   IRR.PASSWORD.RESET CLASS(FACILITY) ID(HELPDESK)  ACCESS(READ)

SETROPTS CLASSACT(FACILITY)
```

Example 2

An administrator with user ID JBAEZ is responsible for maintaining various application user IDs. Often, the passwords for these application user IDs need to be reset. The administrator uses the NOEXPIRED operand of the ALTUSER command so that the new password value does not need to be changed at the next logon of the application user ID. The administrator can be authorized for this task by granting the JBAEZ user ID UPDATE access to the IRR.PASSWORD.RESET resource in the FACILITY class as follows:

```
RDEFINE  FACILITY IRR.PASSWORD.RESET UACC(NONE)
PERMIT   IRR.PASSWORD.RESET CLASS(FACILITY) ID(JBAEZ) ACCESS(UPDATE)

SETROPTS CLASSACT(FACILITY)
```

ALTUSER Examples

Example 1: Resetting a User's Password: A help desk consultant has been authorized to reset passwords. The consultant's RACF user ID (or the RACF group associated with the consultant's user ID) has been permitted by the security administrator with READ access to the IRR.PASSWORD.RESET resource in the FACILITY class. To reset user JIMBOB's password, the consultant enters:

```
ALTUSER JIMBOB PASSWORD(TEMP012X)
```

Example 2: Resetting an Application's Password: A help desk consultant has been authorized to reset passwords. The consultant's RACF user ID (or the RACF group associated with the consultant's user ID) has been permitted by the security administrator with UPDATE access to the IRR.PASSWORD.RESET resource in the FACILITY class. In this example, at the request of operations personnel, the consultant is resetting the user ID associated with an application called CUSTAPP. The consultant uses the NOEXPIRED operand so the application user ID (CUSTAPP in this example) does not need to change the password when it is logged on.

To reset the application's password, the consultant enters:

```
ALTUSER CUSTAPP PASSWORD(VH1SGR8) NOEXPIRED
```

Creating Resource Group Profiles

Like generic profiles, *resource group profiles* enable you to protect multiple resources with one profile. However, the resources do not have to have similar names.

A resource group profile is a general resource profile with the following special characteristics:

- Its name does not match the resources it protects.
- The ADDMEM operand (not the profile name itself) specifies the resources it protects.
- Its class is a resource group class (for example, GTERMINL or GDASDVOL).
- The related member class (not the resource group class itself) must be RACLISTed. For example, the TERMINAL class must be RACLISTed, not the GTERMINL class. Depending on the class, RACLISTing is accomplished using the SETROPTS command or RACROUTE REQUEST=LIST.

For example, the following profile protects three terminals that have *unlike* names, M01RF267, M03RF168, and M04GG148:

```
RDEFINE GTERMINL DEPT35 UACC(NONE) ADDMEM(M01RF267 M03RF168 M04GG148)
```

General Resources

Certain member classes cannot be used with RACF commands because they are associated with resource grouping classes that have special uses. The member classes are GMBR, NODMBR, PMBR, RVARSMBR, SCDMBR, VMBR, and VXMBR. Their associated resource grouping classes are GLOBAL, NODES, PROGRAM, RACFVARS, SECDATA, VMEVENT, and VMXEVENT, respectively.

Table 21 shows the resource group classes and their related member classes.

Table 21. Resource Group Classes

Resource	Resource Group Class	Related Member Class
Terminals	GTERMINL	TERMINAL
DASD volumes	GDASDVOL	DASDVOL
IMS transactions	GIMS	TIMS
Other IMS resources	DIMS and others	CIMS and others
CICS transactions	GCICSTRN	TCICSTRN
CICS tables	HCICSFCT and others	FCICSFCT and others
SDSF authorized commands	GSDSF	SDSF
Cryptographic keys	GCSFKEYS	CSFKEYS
Information/Management resources	GINFOMAN	INFOMAN
Installation-defined classes	Installation-defined class names	Installation-defined class names

To use resource group profiles, take the following steps (terminals are used as a readily understood example):

1. Create the resource group profile:

   ```
   RDEFINE GTERMINL profile-name UACC(NONE)
       ADDMEM(resource-name-with-or-without-generic-character...)
   ```

   where:

   GTERMINL is the resource group class for terminals.

   profile-name is a discrete profile name of your choice (generic characters are not allowed).

   resource-name...
   is the name of the resource to be protected, for example, a terminal ID or DASD volume serial number. If you first activate generic profile checking for the related member class, you can include a generic character (*, **, or %) in the resource name.

2. Grant the appropriate access to the appropriate users and groups. In the following example, READ access is given to users in group GROUPA:

   ```
   PERMIT DEPT35 CLASS(GTERMINL) ID(GROUPA) ACCESS(READ)
   ```

3. When you are ready to start using the protection defined in the profiles, activate the *member class.* For classes other than the CICS[1] and IMS-related classes, you must also activate SETROPTS RACLIST processing for the *member class*.

   For example, for terminals, issue the following command:

   ```
   SETROPTS CLASSACT(TERMINAL) RACLIST(TERMINAL)
   ```

1. The FCICSFCT class is an exception. You can use SETROPTS RACLIST with that class.

Note: Any time you make a change to a GTERMINL profile, you must also refresh SETROPTS RACLIST processing for the TERMINAL class for the change to take effect:

```
SETROPTS RACLIST(TERMINAL) REFRESH
```

For CICS[1] and IMS-related classes, you only need to activate the class (you cannot request RACLIST processing using the SETROPTS command):

```
SETROPTS CLASSACT(TIMS)
```

Note: If an application uses RACROUTE REQUEST=LIST,GLOBAL=YES to RACLIST a class, you can use SETROPTS RACLIST (*classname*) REFRESH to refresh the class. This includes the CICS and IMS classes that can't be RACLISTed with the SETROPTS RACLIST command.

Adding a Resource to a Profile

To add a resource to a profile, issue the RALTER command with the ADDMEM operand, and then refresh the in-storage profiles for that class. For example:

```
RALTER   GTERMINL DEPT35 ADDMEM(M01RF268)
SETROPTS RACLIST(TERMINAL) REFRESH
```

Deleting a Resource from a Profile

To delete a resource from a profile, issue the RALTER command with the DELMEM operand, and then refresh the in-storage profiles for that class. For example:

```
RALTER   GTERMINL DEPT35 DELMEM(M01RF268)
SETROPTS RACLIST(TERMINAL) REFRESH
```

Which Profiles Protect a Particular Resource?

RACF does not prevent you from specifying the same resource in more than one resource grouping profile. If you do so, more than one profile is used to determine the actual protection used. (See "Resolving Conflicts among Multiple Profiles" on page 224.) It can be difficult to determine exactly what protection any one resource has.

To find out if more than one profile protects a particular resource, issue the RLIST command with the RESGROUP operand as follows:

```
RLIST TERMINAL resource-name RESGROUP
```

Make sure to specify the member class (such as TERMINAL or DASDVOL) on the RLIST command. The profiles that protect the terminal appear in the RLIST output under the "RESOURCE GROUPS" heading.

For example, assume that the following commands were issued:

```
RDEFINE GTERMINL DEPT20 ADDMEM(T1 T2 T3)
RDEFINE GTERMINL DEPT22 ADDMEM(T3)
```

To list all of the profiles that protect terminal T3, enter:

```
RLIST TERMINAL T3 RESGROUP
```

In response, RACF displays:

```
RESOURCE GROUPS
-------- ------
DEPT20   DEPT22
```

Note: If a “member class” profile exists for the resource (in this example, if RDEFINE TERMINAL T3 had been issued), the RLIST output includes both the resource groups and the listing of the TERMINAL profile.

Resolving Conflicts among Multiple Profiles

A resource name can appear in more than one resource group and can also have a profile of its own. If a resource is protected by more than one profile, RACF resolves any conflicts by merging the information from the individual profiles. This merging occurs during RACLIST processing according to the following rules:

Rule	Example
The most restrictive UACC is used.	If one profile has a UACC of NONE and another has a UACC of UPDATE, the UACC of NONE is selected.
For any particular user, the least restrictive of the access list entries is used.	For user STEVEH, if one profile has an access list entry of STEVEH(NONE) and another has an access list entry of STEVEH(UPDATE), the access list entry of STEVEH(UPDATE) is selected.
The lowest security level is used.	If one profile has a security level of CONFIDENTIAL and another has the lower security level of ROUTINE, the security level of ROUTINE is selected.
Auditing is done if requested by any of the profiles.	If one profile requests auditing and another does not, auditing is selected.
Category lists are combined.	If one profile has a security category of ACCTG and another profile has a security category of PAYROLL, both the ACCTG and PAYROLL security categories are selected.
The first security label found is chosen.	RACF chooses the security label of the first profile it encounters during RACLIST processing and ignores the security labels for subsequent profiles that must be merged. Therefore, the value of the “merged” security label depends on the order in which the profiles are processed.

If you do not want to use the default rules for merging the information from multiple profiles, you can use the REQUEST=LIST exit routines to change them.

For more information about RACLIST processing and the REQUEST=LIST exit routines, see *z/OS SecureWay Security Server RACF System Programmer’s Guide* and *z/OS SecureWay Security Server External Security Interface (RACROUTE) Macro Reference*.

Because of the way in which profiles are merged, it can be difficult to determine exactly what protection any one resource has. Accordingly, IBM recommends that you do *not* specify the same resource in more than one profile.

How Is WARNING Mode Merged for Conflicting Multiple Profiles?

When a RACLIST is issued and multiple profiles in the grouping class are merged based on their member names, if one of the profiles is in WARNING mode, RACF uses the setting (either WARNING or NOWARNING) for the first profile member or member profile of that name it encounters.

Notes:

1. RACF processes grouping profiles before it processes member profiles.
2. A RACROUTE REQUEST=LIST selection exit can change the results.
3. The FILTER= or LIST= operands of RACROUTE REQUEST=LIST can change the results.

4. RACF ignores WARNING completely unless the issuer of RACROUTE REQUEST=LIST specified RELEASE=1.8 or higher on the macro invocation.

Considerations for Resource Group Profiles

When you work with resource group profiles, keep these considerations in mind:

- There are limitations on the size of resource access lists and profiles, particularly for profiles that are processed in storage by the SETROPTS RACLIST command or the RACROUTE REQUEST=LIST macro. For more information, see “Limiting the Size of Your Access Lists” on page 205.
- Do *not* issue the SETROPTS RACLIST command for the resource group class (for example, GTERMINL or GDASDVOL).

  Instead, specify the related member class (for example, TERMINAL or DASDVOL). When you RACLIST the TERMINAL class, RACF RACLISTs the GTERMINL class for you.
- You *cannot* use the SETROPTS command to RACLIST resource classes for these resources:
  - CICS resources (*except* FCICSFCT)
  - All IMS resources.

  These CICS and IMS resources issue RACROUTE REQUEST=LIST at initialization time.

  To refresh CICS classes that are *not* RACLISTed with RACROUTE REQUEST=LIST,GLOBAL=YES or SETROPTS RACLIST, issue this CICS command from the operator console:

  ```
  CEMT PERFORM SECURITY REBUILD
  ```

  When IMS is refreshed, the IMS classes are refreshed as well.
- You *cannot* specify generic profile names in the resource group class.
- You *can* specify generic names on the ADDMEM operand. However, you should consider defining your generics in the MEMBER class so that the RLIST command can be used to find the generic profile that produces a resource.
- A resource group profile, which is associated with only *one* resource class, *cannot* be used to group resources from two different classes.
- If you use resource grouping profiles, consider avoiding the use of the related member class.

  For example, if you use GTERMINL profiles, convert entirely to using GTERMINL profiles, and delete all TERMINAL profiles. This can ease the administration of terminal authorizations. For example, the SEARCH command lists profile names for only one class at a time: GTERMINL or TERMINAL.

  Note: Remember that you can use RLIST to find the generic that matches a name only if you use member class profiles. RLIST does not provide this support for members of grouping class profiles. Therefore, you must decide which approach is easier to administer. It may be better to define all discrete names as members of grouping profiles and all generic names as member class profiles. That allows you to use multiple SEARCH or RLIST commands when necessary.

  When converting generic TERMINAL profiles to GTERMINL profiles, you can specify generic characters on the ADDMEM operand to obtain the same coverage.

Using RACF Variables in Profile Names (RACFVARS Class)

To keep administrative overhead as low as possible, you should use a single profile to cover multiple resources whenever you can. One way to do this is to define a generic profile, using generic characters in the profile name. Another way is to use a resource grouping class. This section discusses a third way, which is to use RACF variables in the profile name.

Using a RACF variable in a profile name allows you to use one general resource profile to protect many resources with unlike names. Use this method to protect resources whose names are unlike and for which no resource grouping class is available. To protect resources whose names are similar, use profiles with generic characters in their names. To protect resources for which resource grouping classes (for example, GTERMINL and GDASDVOL) are available, use resource grouping classes.

RACF variables can be used for general resource profiles only. You cannot use them in data set profile names.

Profiles whose names contain RACF variables are considered to be generic profiles.

Defining RACF Variables

RACF variable names must begin with an ampersand (&), can be up to eight characters long, and cannot contain any generic characters or periods (.). Do not define variable names that start with &RAC; they are reserved for RACF use.

Several resource names can be assigned to each variable through a profile in the RACFVARS class. The resource names assigned to the variable are added as member names to the RACFVARS profile. The resource names can be up to 39 characters long and cannot contain any generic characters. All of the resources must belong to the same class and must belong to a class that accepts generic profile names.

The UACC of a RACFVARS profile controls who can display the profile to see how a particular variable is defined. A UACC of READ allows anyone to look at the profile using the RLIST command. A UACC of NONE denies the access.

PERMIT commands that are issued for a RACFVARS profile affect the administration of the profile, not access to the resource that is protected with the RACFVARS variable name. For example, to allow users to access tape volumes protected by a TAPEVOL profile called &PAYTAPE, issue a PERMIT command for the profile in the TAPEVOL class, not the profile in the RACFVARS class. However, to allow a user to change the RACFVARS profile called &PAYTAPE, give the user ALTER access authority to the profile in the RACFVARS class, not the profile in the TAPEVOL class.

Because the RACFVARS class requires high performance, you must RACLIST the profiles. To activate the RACFVARS class and RACLIST the profiles, enter:

```
SETROPTS CLASSACT(RACFVARS) RACLIST(RACFVARS)
```

Any time a change is made to a RACFVARS profile, the in-storage profiles for the RACFVARS class must be refreshed for the changes to take effect. To refresh the in-storage profiles for the RACFVARS class, enter:

```
SETROPTS RACLIST(RACFVARS) REFRESH
```

Example of Protecting Several Tape Volumes Using the RACFVARS Class

The easiest way to show how to use the RACFVARS class is by an example.

Suppose you want to protect several tape volumes called TAP111, A22222, and B33OLD, and give ALTER access to them only to a group called PAYGRP. There is no resource grouping class for the TAPEVOL class and the names of the tape volumes are unlike, so you choose the RACFVARS method to protect the tape volumes. To do this, enter:

```
RDEFINE  RACFVARS &PAYTAPE UACC(NONE) ADDMEM(TAP111 A22222 B33OLD)
RDEFINE  TAPEVOL  &PAYTAPE UACC(NONE)
PERMIT   &PAYTAPE CLASS(TAPEVOL) ID(PAYGRP) ACCESS(ALTER)
SETROPTS CLASSACT(TAPEVOL RACFVARS) RACLIST(RACFVARS)
```

If you later need to add a tape volume called C44TAP to the list of protected tape volumes, enter:

```
RALTER   RACFVARS &PAYTAPE ADDMEM(C44TAP)
SETROPTS RACLIST(RACFVARS) REFRESH
```

Using RACF Variables

There are several ways you can use RACF variables. They include:

- Using a RACF variable as the entire name of a profile
- Using a RACF variable as a qualifier in a profile name that has more than one qualifier
- Using the &RACLNDE variable to identify local nodes

Using a RACF Variable as the Entire Name of a Profile

You can use a RACF variable as the entire name of a profile. For example, a TAPEVOL profile name has only one qualifier. It identifies the tape volume to protect. You can create a RACFVARS profile named &PAYTAPE that specifies several tape volumes as members. If you then define a TAPEVOL profile called &PAYTAPE, the profile protects all of the member tape volumes.

Using a RACF Variable as a Qualifier

You can use a RACF variable as a qualifier in a profile name that has more than one qualifier. For example, a JESJOBS profile name identifies the job names to protect. It takes the form SUBMIT.*nodename.jobname.userid* and consists of several qualifiers. You can create a RACFVARS profile named &PROTJOB that specifies several job names as members. If you then define a JESJOBS profile called SUBMIT.*.&PROTJOB.*, the profile protects all of the member job names from any node and any user.

In a profile name, a variable name is ended by the eighth character, the end of the profile name, or one of the following characters, whichever occurs first: a period (.), another ampersand (&), a percent sign (%), or an asterisk (*).

For example, suppose you define the following profile:

```
RDEFINE RACFVARS &ABCDEFG ADDMEM(A B)
```

In this case, X.&ABCDEFGY.Z matches both X.AY.Z and X.BY.Z.

Using the &RACLNDE Profile to Identify Local Nodes

A RACFVARS profile named &RACLNDE can be used to identify node names that are to be treated as local nodes. IBM highly recommends that you use this profile with the NODES, JESJOBS, or JESSPOOL classes. This profile is required for

spool reload functions, starting with JES2 3.1.3 and JES3 3.1.3. For an example, see “Allowing a TSO User to CANCEL All Jobs Originating from Local Nodes” on page 451.

RACFVARS Considerations

To create a RACFVARS member list, use the RDEFINE command with the ADDMEM operand. To add RACFVARS members to an existing member list, use the RALTER command with the ADDMEM operand. To reorder a RACFVARS member list, delete the variable by using RDELETE, and redefine it. To list a RACFVARS member list, use the RLIST command.

Note: The RLIST command lists the members of the RACFVARS in alphabetic order, not in the order entered.

When RACF compares a resource name with a profile name containing RACFVARS, it compares the resource name with each name in the RACFVARS member list. The member names are arranged in the order entered. The oldest member name (the first name in the member list) is checked first and the last member name is checked last. Each character of the resource name is compared with each character of the RACFVARS member name. The search stops at the *first* match of a sequence of characters in the resource name and a RACFVARS member name.

This method of checking can cause unexpected results; for instance, when the member list contains names that are a subset of other members in the list. In this case, the resource name does not match the expected profile name.

The following three examples point out some important factors to consider when working with RACFVARS.

Example 1

```
RDEFINE RACFVARS &CTM ADDMEM(TEST TESTA)
RDEFINE SURROGAT &CTM.SUBMIT UACC(NONE)
PERMIT  &CTM.SUBMIT CLASS(SURROGAT) ID(USER1) ACCESS(READ)
```

The job TESTA1 submitted by USER1 on system PLPSC, with USER=TESTA on the job card, results in a failure and the following error message.

```
$HASP165  TESTA1 ENDED AT PLPSC - SECURITY VIOLATION
```

The failure occurs because RACF checking stops when the first four characters of the specified resource name, TESTA, match the first RACFVARS member, TEST, leaving the letter A. The remaining letter A is considered a specific part of the resource name and there is no corresponding specific part in the profile name to which it can be matched.

As a precaution, when adding RACFVARS members, order the member names. The member names that are a subset of other names should follow the names of which they are a subset.

In the preceding example, TEST is a subset of TESTA. Therefore, to obtain the expected result, reverse the members in the RACFVARS member list.

```
RDEFINE RACFVARS &CTM UACC(NONE) ADDMEM(TESTA TEST)
```

Note: Ordering the members solves the problem in the previous example. However, this may not be the desired order in all cases.

Example 2

```
RDEFINE RACFVARS &R ADDMEM(AB A)
RDEFINE ACCTNUM  &R%.X UACC(NONE)
PERMIT  &R%.X CLASS(ACCTNUM) ID(USER1) ACCESS(READ)
```

In this example, TSO user USER1 attempts to log on with account number AB.X, but profile &R%.X does not match. This results in the following error message:

```
IKJ56486I  THE ACCOUNT NUMBER AB.X HAS NOT BEEN DEFINED FOR USE
```

The AB matches appropriately. However, no characters remain in the resource name to match with the generic character, %.

To obtain the expected result, reverse the members in the RACFVARS member list as follows:

```
RDEFINE RACFVARS &R ADDMEM(A AB)
```

or redefine the generic profile as follows:

```
RDEFINE ACCTNUM &R*.X UACC(NONE)
```

When you use any of the following to define a profile name, unexpected results can occur:

- Multiple RACFVARS
- A combination of RACFVARS and generic characters
- A combination of RACFVARS and specific names

Example 3

```
RDEFINE SURROGAT &A&B.SUBMIT UACC(NONE)
PERMIT  &A&B.SUBMIT CLASS(SURROGAT) ID(USER1) ACCESS(READ)
RDEFINE RACFVARS &A UACC(NONE) ADDMEM(AB A)
RDEFINE RACFVARS &B UACC(NONE) ADDMEM(B C)
```

The job AB1 submitted by USER1 on system PLPSC, with USER=AB on the job card, results in a failure and the following error message:

```
$HASP165  AB1 ENDED AT PLPSC - SECURITY VIOLATION
```

The failure occurs because RACF checking for the resource name, AB, matches the first member of &A, which is AB. Because there is no part of the resource name to match the second part of the profile name specified by &B, the compare fails.

The resource name must match with a member of each of the RACFVARS used to define a profile.

To obtain the expected results, reverse the members in the RACFVARS member list of &A:

```
RDEFINE RACFVARS &A UACC(NONE) ADDMEM(A AB)
```

However, the set of resource names that was valid has now changed. For example, the specific resource name, ABB, was valid and is no longer valid.

In summary, to avoid unexpected results, reduce the complexity of profiles.

Using RACFVARS with Mixed-Case Classes

Using RACF variables in mixed-case profile names has limited use. A mixed-case profile is a profile in a class defined in the class descriptor table (CDT) with the CASE=ASIS option, and may, therefore, have a name that contains lowercase characters. The RACFVARS class itself is not defined with this option, so its profiles and member values will be treated as uppercase. Therefore, when you define a mixed-case profile using a RACF variable, you must enter the variable name in uppercase.

In the following example, $MYCLASS is a mixed-case class. Notice that the profile called My.Pet.&VAR contains mixed-case characters but the variable name (&VAR) is uppercase. This allows the variable name match the RACFVARS profile name, which must be uppercase.

```
RDEFINE RACFVARS &VAR ADDMEM(CAT DOG FISH)
RDEFINE $MYCLASS My.Pet.&VAR
```

Because the &VAR profile can contain only members with uppercase names (CAT, DOG, and FISH), a resource named My.Pet.FISH is covered by these profiles, but My.Pet.Fish is not.

Note: Do not enter profile name My.Pet.&var because, although RACF will accept the name, the character string &var will not match the RACFVARS profile name &VAR and will not be recognized as a RACF variable.

Controlling VTAM LU 6.2 Bind

With VTAM 3.3 or later, and with a product like CICS 3.2 or later, you can control which type 6.2 logical units can establish sessions with each other. This includes the ability to use RACF to specify the values used in password-on-bind processing. For more information, see "RACF and APPC" on page 292, *z/OS Communications Server: SNA Programmer's LU 6.2 Guide*, or *z/OS MVS Planning: APPC/MVS Management*.

To do this, take the following steps:

1. Ask your VTAM system programmer for the following information for each VTAM LU 6.2 pair:
   - The network ID and the LU identifiers for each member of the pair.
   - Whether or not a password is required for session verification based on the VERIFY option specified on the VTAM APPL statement for the LU in SYS1.VTAMLST. (This password is referred to as the *session key* in your RACF definitions.)
   - Whether or not the NQNAME option is specified for the ACB. If it is specified, this indicates that network-qualified names support is enabled.
2. For each LU 6.2 pair, create two profiles in the APPCLU class.

   On one system, enter one of the following RDEFINE commands. If network-qualified names support is *not* enabled, enter:

   ```
   RDEFINE APPCLU local-netid.luid1.luid2 UACC(NONE)
   ```

   If network-qualified names support *is* enabled, enter:

   ```
   RDEFINE APPCLU local-netid.luid1.remote-netid.luid2 UACC(NONE)
   ```

   On the other system, enter one of the following RDEFINE commands. If network-qualified names support is *not* enabled, enter:

   ```
   RDEFINE APPCLU local-netid.luid2.luid1 UACC(NONE)
   ```

   If network-qualified names support *is* enabled, enter:

   ```
   RDEFINE APPCLU local-netid.luid2.remote-netid.luid1 UACC(NONE)
   ```

   where:

   local-netid, remote-netid
   : are the network IDs (NETID) of the partners. These IDs are specified on the VTAM start option NETID, which is in the ATCSTR*xx* member of SYS1.VTAMLST.

   luid1, luid2
   : are the LU names of the partners. In each case, the first LU name specified is the local LU name, and the second LU name is the partner LU name.

   For each profile created, the first LU name specified (*luid1*) is the *primary LU* on that system.

   Note: You should not specify an asterisk (*) or any other generic character for any of the *netid* and *luid* qualifiers.
3. Define the attributes of the sessions between the partners of each LU pair. You do this by defining a SESSION segment for each APPCLU profile using the SESSION option of the RDEFINE and RALTER commands. You can specify the following information in each SESSION segment:

CONVSEC Specifies the level or levels of security checking performed for each conversation between the partners of the LU pair.

INTERVAL Specifies the maximum number of days the session key is valid before it must be changed.

LOCK Indicates that a session between the partners of the LU pair cannot be established.

SESSKEY(*session-key***)**
Specifies the password used to verify sessions between the partners of this LU pair. If specified, the SESSKEY value must be the same in both APPCLU profiles for this LU pair. A session key may be required based on the VERIFY option specified on the VTAM APPL statement for this LU pair.

Note: Session keys are 64-bit data encryption standard (DES) keys. With 64-bit DES encryption, 8 of the 64 bits are reserved for use as parity bits. This means that those 8 bits are not part of the 56-key. In hexadecimal notation, the DES parity bits are: X'0101 0101 0101 0101'. Therefore, any two 64-bit keys are equivalent if their only difference is in one or more of these parity bits. For example, the following SESSKEY values, although appearing to be quite different, are equivalent because they differ only in the last bit of each byte:

```
BDF0KM4Q (X'C2C4 C6F0 D2D4 F4D8'
CEG1LN5R (X'C3C5 C7F1 D3D5 F5D9'
```

For detailed information about using the RDEFINE and RALTER commands to define options in the SESSION segment, see *z/OS SecureWay Security Server RACF Command Language Reference*.

4. Optionally, for maintenance purposes, give users and groups appropriate access authority:

   ```
   PERMIT profile-name CLASS(APPCLU) ID(userid or groupname)
      ACCESS(access-authority)
   ```

   where *access-authority* is one of the following:

   NONE Allows no access to the profile

   READ Allows users to list the profile (for example, using the RLIST and SEARCH commands)

   UPDATE Is the same as READ

   CONTROL Is the same as READ

   ALTER Allows users to change the profile (if the profile is discrete)

5. When you are ready to start using the protection defined in the profiles, activate the APPCLU class on *every* system on which you want to use the APPCLU profiles:

   ```
   SETROPTS CLASSACT(APPCLU)
   ```

   Note: You cannot issue the SETROPTS RACLIST command for the APPCLU class. VTAM does this for you (using RACROUTE REQUEST=LIST).

Example:

Suppose there are two large nodes, one in New York and the other in Tokyo.

Network-qualified names support is *not* enabled.

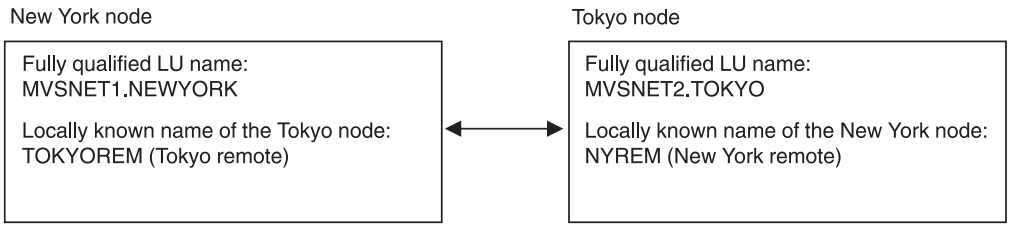


Figure 9. Example of Two Network LU Partners

On the New York node, take the following steps:

1. Define a profile for the Tokyo LU partner:
   ```
   RDEFINE APPCLU MVSNET1.NEWYORK.TOKYOREM SESSION(SESSKEY(KEY1)) UACC(NONE)
   ```
2. Activate the APPCLU class:
   ```
   SETROPTS CLASSACT(APPCLU)
   ```

On the Tokyo node, take the following steps:

1. Define a profile for the New York LU partner:
   ```
   RDEFINE APPCLU MVSNET2.TOKYO.NYREM SESSION(SESSKEY(KEY1)) UACC(NONE)
   ```
2. Activate the APPCLU class:
   ```
   SETROPTS CLASSACT(APPCLU)
   ```

Protecting Applications

You can use a combination of a profile in the APPL class and the APPL operand on the RACROUTE REQUEST=VERIFY macro to control which users can use an application as they enter the system.

To do this, take the following steps:

1. Determine the name of your application.
2. Specify the name of the application on the APPL operand of the RACROUTE REQUEST=VERIFY macro.
3. Create a profile in the APPL class:
   ```
   RDEFINE APPL applname UACC(NONE)
   ```
4. Give users and groups READ access, as appropriate.
   ```
   PERMIT applname CLASS(APPL) ID(userid or groupname) ACCESS(READ)
   ```
5. If you have not already done so, activate the APPL class:
   ```
   SETROPTS CLASSACT(APPL)
   ```
6. For performance reasons, request SETROPTS RACLIST or SETROPTS GENLIST processing for the APPL class.

   Note: This may be important if many users enter the system under control of your application (where your application issues the RACROUTE REQUEST=VERIFY macro for each user).

For information on how authorization checking takes place, see “Authorizing Access to RACF-Protected Applications” on page 647.

Protecting DFP-Managed Temporary Data Sets

You can protect DFP-managed temporary data sets. Normally, these data sets are considered protected from any accesses except by the job or session that created them, and therefore do not need to be protected by RACF. However, the following situations could leave a temporary data set unprotected:

- A system failure
- An initiator failure or initiator termination by the FORCE command
- An automatic restart—between the failure and the restart

In these cases, if the TEMPDSN class is active, only users with the OPERATIONS attribute can scratch any residual DFP-managed temporary data sets remaining on a volume.

Note: The user with the OPERATIONS attribute can access the data set only to scratch the data set. No other access is allowed (such as would be allowed by READ or UPDATE access authority to the data set).

To activate the TEMPDSN class, enter:

```
SETROPTS CLASSACT(TEMPDSN)
```

Avoid activating the TEMPDSN class when current users or jobs are using temporary data sets. Otherwise, you could cause users or jobs to receive an ABEND, as shown in the following scenario:

1. The job or user allocates a temporary data set.
2. You activate the TEMPDSN class.
3. The job or user opens the data set.
4. Because activating the TEMPDSN class restricts the authority to open a temporary data set, the user or job receives an abend.

Protecting File Services Provided by LFS/ESA

If LAN File Services/ESA (LFS/ESA) Release 1 is installed, you can use the LFSCLASS class to control user access to LAN File Services.

The resource name contains two parts:

- A data set name
- A workstation directory name

The two parts are separated with a colon. The workstation directory name contains the directory name and at least one subdirectory, with a backslash(\) before every subdirectory name. A maximum length of 246 characters is supported.

Here is an example of using the LFSCLASS class:

- Create a profile in LFSCLASS class:

  ```
  RDEFINE LFSCLASS MVS.DATASET.NAME:\DIR1\SUBDIR
  ```

- If you have not already done so, activate the LFSCLASS class:

  ```
  SETROPTS CLASSACT(LFSCLASS)
  ```

Note: RACF supports the backslash(\) character for use with the LFSCLASS class. However, you may need to change your VTAM translation table to support the backslash(\). For more information, refer to *z/OS TSO/E Programming Guide* for your operating system.

Protecting Terminals

There are several methods of controlling the use of terminals that are connected to your system. The following sections describe these methods:

- "Creating Profiles in the TERMINAL and GTERMINL Classes". You must give users at least READ access authority in order to allow them to use protected terminals. *You must do this before using any of the other methods for controlling terminals.*
- "Controlling the Use of Undefined Terminals" on page 236. By specifying TERMINAL(NONE) on the SETROPTS command, you can prevent users from logging on to terminals unless the terminals are protected by profiles in the TERMINAL or GTERMINL classes.

  Attention
  Do not protect undefined terminals unless you have created profiles that allow users to access the terminals they currently use.

- "Limiting Specific Groups of Users to Specific Terminals" on page 237. By specifying NOTERMUACC on the ADDGROUP or ALTGROUP command, you can restrict users in those groups to terminals whose access lists specifically allow the user or the user's group to use the terminal.
- "Limiting the Times That a Terminal Can Be Used" on page 238. By specifying the WHEN operand on the RDEFINE and RALTER commands for profiles in the TERMINAL and GTERMINL classes, you can specify the days and times that users can log on to terminals.
- "Using Security Labels to Control Terminals" on page 238. If the SECLABEL class is active, you can control access to terminals by specifying security labels for profiles in the TERMINAL and GTERMINL classes.
- "Using the TSO LOGON Command with the RECONNECT Operand" on page 238. TSO allows verification and checking so that a user can resume an interrupted session from a new terminal.

For a description of authorization checking for terminals, see "Authorizing Access to RACF-Protected Terminals" on page 644.

Creating Profiles in the TERMINAL and GTERMINL Classes

If you create a profile in the TERMINAL or GTERMINL class, you must give users at least READ access authority in order to allow them to use the protected terminal.

1. To protect a terminal using RACF, create a profile for it using the RDEFINE command. On the command, specify the universal access authority (UACC) you want to assign to the terminal. The following command defines a profile for terminal M01RF267 and specifies a UACC of NONE.

   ```
   RDEFINE TERMINAL M01RF267 UACC(NONE)
   ```

   On systems using VTAM or TCAM, the terminal's node name is the RACF resource name. On systems using BTAM, the resource name for the terminal consists of the relative line and terminal number. This information can be obtained from the system programmer who is responsible for the Network Communication Program (NCP) system generations for your installation.
2. Use the PERMIT command to allow users and groups to use the terminal. You must give a user at least READ access authority to the terminal. Otherwise, the

user is not authorized to use the terminal. For example, the following command grants users SMITH and JONES READ access authority to terminal M01RF627.

```
PERMIT M01RF267 CLASS(TERMINAL) ID(SMITH JONES) ACCESS(READ)
```

Attention
After you define a terminal and protect it with a UACC of NONE, no one can use the terminal until you grant users or groups READ access authority to the resource.

3. When you are ready to start using the protection defined in the profiles, activate the TERMINAL class. You should also consider activating SETROPTS RACLIST processing for the class. SETROPTS RACLIST processing helps ensure high performance when access authorities are checked. Also, if you are using GTERMINL profiles, you *must* request RACLIST processing for the TERMINAL class. You can do these two actions in one command:

   ```
   SETROPTS CLASSACT(TERMINAL) RACLIST(TERMINAL)
   ```

   Note: When you activate the TERMINAL class, RACF also activates the GTERMINL class.

Creating a Profile in the GTERMINL Class: If you want to protect several terminals in the same way, but their names do not allow you to create a generic profile, you can create a profile in the GTERMINL class for them. For example, to protect terminals M01RF267, M03RF168, and M04GG148 with one profile, you could create a profile with a name you choose, such as DEPT35:

```
RDEFINE GTERMINL DEPT35 UACC(NONE) ADDMEM(M01RF267 M03RF168 M04GG148)
```

To allow group FINANCE to use these terminals, enter:

```
PERMIT DEPT35 CLASS(GTERMINL) ID(FINANCE) ACCESS(READ)
```

Note: After creating or changing a GTERMINL profile, you must request SETROPTS RACLIST processing for the TERMINAL class to make the changes effective on the system.

To protect another terminal, named M01RF299, with the same profile, change the DEPT35 profile as follows:

```
RALTER   GTERMINL DEPT35 ADDMEM(M01RF299)
SETROPTS RACLIST(TERMINAL) REFRESH
```

To stop protecting terminal M03RF168 with this profile, change the DEPT35 profile as follows:

```
RALTER   GTERMINL DEPT35 DELMEM(M03RF168)
SETROPTS RACLIST(TERMINAL) REFRESH
```

Controlling the Use of Undefined Terminals

You can also use RACF to control the use of undefined terminals that are connected to your system. To control the use of undefined terminals, you must first activate the TERMINAL class as shown above. After the TERMINAL class is active, you can control whether users can log on to undefined terminals by issuing the SETROPTS command with the TERMINAL operand. The TERMINAL operand specifies the universal access authority, either READ or NONE, that RACF associates with undefined terminals on your system.

To allow undefined terminals to be used for logging on, enter:

```
SETROPTS TERMINAL(READ)
```

To prevent undefined terminals from being used for logging on, enter:

```
SETROPTS TERMINAL(NONE)
```

Attention

Before you specify NONE, be sure that you define some terminals to RACF and give the appropriate users and groups proper authorization to use them. Otherwise, *no one can log on to your system*.

Combining the SETROPTS TERMINAL Command with TERMINAL Profiles

If you want to control selected terminals, specify READ on the TERMINAL operand of the SETROPTS command. When you specify READ, all users can access all terminals. To control access to selected terminals, define each terminal individually and specify a UACC of NONE. Then, create an access list for each terminal that contains the user IDs of the users who require access to the terminal.

If you decide that you want to control *all* terminals, specify NONE on the TERMINAL operand of the SETROPTS command. When you specify NONE, only users and groups that you authorize to use a terminal through its access list can use it.

Attention

Before you specify NONE, be sure that you define some terminals to RACF and give the appropriate users and groups proper authorization to use them. Otherwise, *no one can log on to your system*.

Limiting Specific Groups of Users to Specific Terminals

When defining or changing a group profile, you can specify that the group can log on only to those terminals to which the group (or individual users within the group) are specifically authorized. If the group terminal option NOTERMUACC is in effect (note that TERMUACC is the default) for a group on the ADDGROUP or ALTGROUP command, users of the group can use only those terminals to which they are specifically authorized on the access list in the TERMINAL profile protecting the terminal.

For example, if you want to allow group PAYROLL to log on only to terminals in the payroll office, protect the payroll terminals with a profile:

```
RDEFINE GTERMINL PAYTERMS ADDMEM(M02RF001 M11RF203) UACC(NONE)
```

Give the PAYROLL group READ access:

```
PERMIT PAYTERMS CLASS(GTERMINL) ID(PAYROLL) ACCESS(READ)
```

Ensure that the PAYROLL group profile has NOTERMUACC specified:

```
ALTGROUP PAYROLL NOTERMUACC
```

This prevents users in group PAYROLL from logging on to another terminal just because the profile protecting that terminal has a UACC of READ.

Note: If the list-of-groups option (SETROPTS GRPLIST) is in effect, RACF uses the TERMUACC/NOTERMUACC option from the user's current connect group, but RACF can grant terminal access through any of the user's connect groups.

Limiting the Times That a Terminal Can Be Used

RACF allows you to limit the use of specific terminals to certain days of the week and certain hours of the day. To control when the system can be accessed from the terminal, use the WHEN operand on the RDEFINE and RALTER commands for the TERMINAL or GTERMINL classes. For more information on the time and day-of-week controls, see "Limiting When a User Can Access the System" on page 85 or the command descriptions in *z/OS SecureWay Security Server RACF Command Language Reference*.

For example, to allow logons at a terminal only between 7 a.m. and 5 p.m. during the week, specify WHEN(DAYS(WEEKDAYS) TIME(0700:1700)) on the RDEFINE or RALTER command.

Using Security Labels to Control Terminals

If both the TERMINAL and the SECLABEL class are active, RACF checks a user's authority to use a terminal. To use the terminal, the user must log on with a security label that is less than or equal to the security label of the terminal.

You can use this to limit the sensitivity of the data that users can access from the terminal. For example, if you have some terminals that can be accessed easily by many users, you can assign those terminals a low-sensitivity security label, such as SYSLOW. This prevents users from using those terminals to access data that has a security label higher than the terminal's security label.

Using the TSO LOGON Command with the RECONNECT Operand

TSO provides a line drop facility that enables a user to log on to TSO from another terminal and reconnect to the existing session by issuing the LOGON command with the RECONNECT operand.

During this logon to the new terminal, LOGON command processing requests that RACF verify the user's user ID and password (or operator identification card). Also, if the TERMINAL class is active, the user's authority to access the new terminal is checked. Note that the user cannot change connect groups when the RECONNECT operand is used.

If verified and authorized, the user can resume the interrupted session from the new terminal.

Protecting Consoles

You can require operators to log onto and log off from MCS-managed consoles by specifying options in the CONSOL*xx* member of the SYS1.PARMLIB data set. For more information, see:

- *z/OS MVS Initialization and Tuning Reference*
- *z/OS MVS System Commands*
- *z/OS MVS Planning: Operations*

When the CONSOLE class is active and a console being used is protected by a profile in the CONSOLE class, RACF ensures that the person attempting to logon

has the proper authority to do so. Using RACF, you can control the use of JES and MCS system consoles on your system. This section describes how to protect MCS consoles. See also “Remote Workstations (RJP/RJE Consoles)” on page 478.

Notes:

1. The SETROPTS TERMINAL command does not apply to consoles.
2. The TERMUACC operand on the ADDGROUP and ALTGROUP commands does not apply to consoles.
3. You cannot specify the WHEN operand on the RDEFINE and RALTER commands for profiles in the CONSOLE class.

For a description of authorization checking for consoles, see “Authorizing Access to RACF-Protected Consoles, JES Input Devices, or APPC Partner LUs” on page 645.

To control the use of MCS consoles, take the following steps:

1. Ask your system programmer for the following information:
   - The ID of the console to be protected
   - The universal access authority (UACC) to specify for the console
   - The operator ID or group name of the operators to whom you want to grant access
   - The security label to be assigned to that console (if security labels are being used)
2. Create a profile for each console using the RDEFINE command:

   ```
   RDEFINE CONSOLE console-id UACC(NONE)
   ```

   For example, the following command defines a profile for console 22 and specifies a UACC of NONE.

   ```
   RDEFINE CONSOLE 22 UACC(NONE)
   ```
3. If console logon is required or automatic as specified in the CONSOL*xx* member of SYS1.PARMLIB, use the PERMIT command to allow users and groups to use the console. You must give a user at least READ access authority to the console. Otherwise, the user is not authorized to use the console.

   For example, the following command grants group OPRGRP1 and user JONES READ access authority to console 22.

   ```
   PERMIT 22 CLASS(CONSOLE) ID(OPRGRP1 JONES) ACCESS(READ)
   ```

   Attention

   After you define a console and protect it with a UACC of NONE, no one can log on to the console until you grant users access authority to the console profile.

   For consoles, the valid access authorities are:

   NONE Allows no access

   READ Authorizes RACF-defined users to LOGON to the specified console
4. When you are ready to start using the protection defined in the profiles, activate the CONSOLE class and activate SETROPTS RACLIST processing for the class. SETROPTS RACLIST processing helps ensure high performance when access authorities are checked. You can do these two actions in one command:

   ```
   SETROPTS CLASSACT(CONSOLE) RACLIST(CONSOLE)
   ```

Using Security Labels to Control Consoles

If both the CONSOLE and the SECLABEL class are active, RACF checks an operator's authority to use a console. To use the console, the operator must log on with a security label that is less than or equal to the security label of the console.

Using the Secured Signon Function

If your installation includes workstations and client machines that are operating in a client/server environment, you may want to use the RACF secured signon function to provide enhanced security across a network. The secured signon function provides an alternative to the RACF password called a PassTicket, which allows workstations and client machines to communicate with a host without using a RACF password.

The secured signon function removes the need to send RACF passwords across the network and allows you to move the user authentication part of signing on to a host from RACF to another product or function. End users of an application can use the PassTicket to authenticate their user IDs and log on to computer systems that contain RACF.

This section describes the PassTicket and how to set up the secured signon environment. It includes information about:

- Activating the PTKTDATA class
- Defining profiles in the PTKTDATA class
- The process RACF uses to validate a password or PassTicket
- Enabling the use of PassTickets

For information about the programming that is needed for an application to generate a PassTicket, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

The RACF PassTicket

The RACF PassTicket is a *one-time-only*[2] password that is generated by a requesting product or function. It is an alternative to the RACF password that removes the need to send RACF passwords across the network in clear text. It makes it possible to move the authentication of a mainframe application user ID from RACF to another authorized function executing on the host system or to the workstation local area network (LAN) environment.

Activating the PTKTDATA Class

Before you can use the secured signon function, you must activate the PTKTDATA class. The PTKTDATA class is the class to which all profiles that contain PassTicket information are defined. To activate the class and the function, enter:

```
SETROPTS CLASSACT(PTKTDATA)
SETROPTS RACLIST(PTKTDATA)
```

2. Because it only gives one user access to a specific application for approximately 10 minutes, a RACF PassTicket is resistant to reuse. For most applications, once a particular PassTicket is used, the same user cannot use it again for the same application during the same 10-minute interval.

   By keeping a copy of all used valid PassTickets for the duration of the 10-minute interval during which they might possibly be used again, RACF provides another level of protection against reuse. For performance reasons, RACF uses main memory for this storage. If an application can run on more than one computer with individual memory at the same time, this level of reuse protection might not be available.

After you activate the PTKTDATA class, you can define the necessary profiles.

Note: After you define or change the profiles, you need to refresh the class by entering `SETROPTS RACLIST (PTKTDATA) REFRESH.`

Defining Profiles in the PTKTDATA Class

For each application that users can gain access to with the PassTicket, you must create at least one profile in the PTKTDATA class. The profile associates a secret secured signon application key with a particular application on a particular system. The profiles can be created so they apply to:

- All users who need access to the application
- A specific RACF group of users who need access to the application
- A specific RACF user, when connected to a specific RACF group
- A specific RACF user

To define the profile, use the RDEFINE command:

```
RDEFINE PTKTDATA profile-name
        SSIGNON(key-description)
        UACC(access-authority)
```

where:

PTKTDATA
specifies the PassTicket key class.

profile-name
is the name of the profile (see “Determining Profile Names” on page 242).

For the PTKTDATA class, the profile must be a discrete profile. Because each application must be uniquely defined, you cannot specify a generic profile in the PTKTDATA class. If you specify a generic profile, it is ignored during PassTicket processing for the application, and PassTickets cannot be used to authenticate users for that application.

key-description
defines the secured signon application key and specifies the method RACF is to use to protect it in the RACF database on the host. You can specify either masking or encryption for the method (see “Protecting the Secured Signon Application Keys” on page 244).

Secured signon keys are 64-bit Data Encryption Standard (DES) keys. With DES, 8 of the 64 bits are reserved for use as parity bits, so those 8 bits are not part of the 56-bit key. In hexadecimal notation, the DES parity bits are: X'0101 0101 0101 0101'. Any two 64-bit keys are equivalent DES keys if their only difference is in one or more of these parity bits.

access-authority
is the universal access authority to be associated with the resource protected by this profile. By default, the UACC is NONE for the PTKTDATA class.

After a profile in the PTKTDATA class has been created, you can change it with the RALTER command, which is similar in syntax to the RDEFINE command:

```
RALTER PTKTDATA profile-name
       SSIGNON(key-description)
       UACC(access-authority)
```

Determining Profile Names

A PTKTDATA class profile name can consist of *one* of the following:

- An application name only
- An application name appended (or qualified) by a RACF connect group name
- An application name qualified by a RACF user ID
- An application name qualified by both a RACF connect group name and a RACF user ID.

When the profile name consists of the application name and one or two qualifiers, the qualifiers are separated by a period. When a RACF connect group name and a RACF user ID are used as qualifiers, the group name must be appended to the application name and the user ID must be appended to the group name.

According to this rule, the name structures in the following list can be used as profile names in the PTKTDATA class. Any other name structures will be ignored. In this example, the application name is TSO1234, the user's current connect group name is SYS1, and the user ID is IBMUSER:

1. An application name concatenated with a RACF group name and user ID: TSO1234.SYS1.IBMUSER
2. An application name concatenated with a RACF user ID: TSO1234.IBMUSER
3. An application name concatenated with a RACF group name: TSO1234.SYS1
4. An application name: TSO1234

When PassTicket evaluation occurs, multiple profiles can exist that fit the particular application, user, and group specification. When multiple profiles exist, RACF processing is as follows:

1. Assuming there is at least one qualified profile, RACF selects one qualified profile name according to the precedence shown in the previous list (items 1, 2, and 3).

   The first qualified profile found using this search precedence is selected and RACF evaluates the PassTicket using this key. Any other profiles with qualified names are ignored.
2. If no qualified name is found, or the evaluation using the key within the qualified profile is not successful because the key is not correct, RACF searches for a profile using only the application name. If such a profile exists, RACF evaluates the PassTicket using the key contained within this profile.

Depending on the application (APPC, CICS, IMS, MVS batch, TSO, or VM), the secured signon function uses a specific method for determining profile names in the PTKTDATA class.

Note: Check with your system programmer to see if your installation is using RACF exit ICHRIX01 to modify the application name that RACF uses during user verification processing. If so, the application name used to determine the PTKTDATA class profile name for APPC, CICS, IMS, MVS batch, TSO, or VM applications *must* match the application name ICHRIX01 selects.

For example, if the ICHRIX01 exit places the character string `TSO1234` in the application name position of the exit parameter list, the application name position of the PTKTDATA class profile must also be `TSO1234`.

APPC, CICS, or IMS: To define a profile for an APPC, CICS, or IMS application, define the profile to the PTKTDATA class with a left-most qualifier that matches the standard naming conventions you use to define these applications to the APPL class.

- For general information on the APPL class, see "Protecting Applications" on page 233.
- For APPC information, see "RACF and APPC" on page 292.
- For CICS information, see *CICS RACF Security Guide*.
- For IMS information, see "Chapter 14. RACF and Information Management System (IMS)" on page 425.

MVS Batch Jobs: For MVS batch jobs that include RACF passwords in the job control language (JCL), you can replace the password with a PassTicket. To define a profile for an MVS batch job:

1. Ask the system programmer for the SMF system identifier of the MVS system on which the application is running. It is located in SMFPRM*xx* member of SYS1.PARMLIB and is specified by the SID value.
2. Determine the left-most qualifier name of the profile to represent the MVS batch job application to the PTKTDATA class by using the characters `MVS` as a prefix to the system's SMF identifier.

Example: Creating an MVS Batch Job Profile Name: The SMF identifier of the system the MVS batch job application is running on is SYS2. To create the profile name, use `MVS` as the prefix. The resulting profile name is `MVSSYS2`.

If the profile applies only to RACF users connected to the RACF group PROD, the resulting profile name is `MVSSYS2.PROD`.

Note: If the SMF identifier contains blanks or characters that are not alphanumeric, they cannot be specified in the profile name. For example, if the SMF identifier is `SY*6`, you must specify the profile defined to the PTKTDATA class as `MVSSY6`.

TSO: Ask the system programmer if a VTAM generic resource name for TSO is being used, then reference the appropriate information below.

Creating a TSO Profile Name (when a VTAM Generic Resource Name for TSO is Used): If VTAM generic resource naming is used for TSO application names, ask the system programmer for the generic name for TSO, and use it as the left-most qualifier of the PTKTDATA profile name.

The VTAM generic resource name for TSO is located in:

- The GNAME value in the TSOKEYxx member of SYS1.PARMLIB.
- The `GNAME=` operand of the START command issued to start TSO.

Example: The VTAM generic resource name for TSO on your system is TSOGR, and a PTKTDATA profile needs to be created for the TSO application. Use the VTAM generic resource name as the profile name. The resulting profile name is TSOGR. If the profile applies only to RACF users connected to the RACF group `PROD`, the resulting profile name is `TSOGR.PROD`.

Notes:

1. A VTAM generic resource name allow the name by which an application is known to its end users to be different from the actual application name on a given execution system. This allows multiple real application servers to be used

by large numbers of users who request the services of the application name by its ″generic″ name, while the requested service is actually provided by multiple back-end application servers. With VTAM generic resources, the real back-end application name does not need to be exposed to end users, since they refer to the application only by its generic name.

2. During TSO signon PassTicket evaluation, RACF checks the VTAM terminal address space control table (TCAST) for a VTAM generic resource name for each TSO application environment. If a VTAM generic resource name exists for this particular TSO application, it is used as the application name by RACF for the evaluation process. This results in consistency of application names between PassTicket generation time and evaluation time.

Creating a TSO Profile Name (when a VTAM Generic Resource Name for TSO is Not Used): If VTAM generic resource naming is not used for TSO, you should:

1. Ask the system programmer for the SMF system identifier of the MVS system where the application is running.

   The SMF system identifier is located in the SMFPRM*xx* member of SYS1.PARMLIB and is specified by the SID value.
2. Determine the left-most qualifier of the profile name to represent the TSO application in the PTKTDATA class by using the characters `TSO` as a prefix to the system's SMF identifier.

Example: The SMF identifier of the system the TSO application is running on is `SYS2`. To create the profile name, use `TSO` as the prefix. The resulting profile name is `TSOSYS2`. If the profile applies only to RACF users connected to the RACF group `PROD`, the resulting profile name is `TSOSYS2.PROD`.

Note: If the SMF identifier contains blanks or characters that are not alphanumeric, they cannot be specified in the profile name. For example, if the SMF identifier is `SY-5`, you must specify the profile defined to the PTKTDATA class as `TSOSY5`.

VM Logon: If you are sharing a database with a VM system, you can define a profile for VM:

1. Ask the system programmer for the system ID of the VM system. It can be located by examining the CPU-ID (system ID) field in the RACF SMF CONTROL file.
2. Determine the left-most qualifier name of the profile to represent VM to the PTKTDATA class by using the characters `VM` as a prefix to the CPU-ID field.

Example: Creating a VM Profile Name: The system ID of the VM system is `ISGR8`. To create the profile name, use `VM` as the prefix. The resulting profile name is `VMISGR8`.

If the profile applied only to RACF users connected to the RACF group `PROD`, the resulting profile name would be `VMISGR8.PROD`.

Note: If the CPU-ID field contains blanks or characters that are not alphanumeric, they cannot be specified in the profile name. For example, if the CPU-ID field contains `VM 7`, you must specify the profile defined to the PTKTDATA class as `VMVM7`.

Protecting the Secured Signon Application Keys

When you define the secured signon application keys, RACF either masks or encrypts each key. If the system has a cryptographic product installed and

available, you can encrypt the secured signon application keys for added protection. For more information, see “Masking the Secured Signon Application Key” and “Encrypting the Secured Signon Application Key”.

Note: To prevent unauthorized users from looking at the secured signon application keys that are stored in the RACF database, make sure the universal access authority (UACC) of the RACF database is NONE. This prevents unauthorized users from listing or copying the RACF data set that contains these sensitive keys.

Masking the Secured Signon Application Key: If the system using the secured signon function does not use a cryptographic product, RACF masks the key with a proprietary masking algorithm when you define or alter it. The masking algorithm that masks the secured signon application keys while they reside in the RACF database is an IBM proprietary algorithm. It is designed to provide protection against casual viewing of the secured signon masked keys. The algorithm is *not* a cryptographic algorithm and cannot provide the level of security for the secured signon application keys that the use of cryptography can provide.

To mask the secured signon application key when you define or alter it, use the SSIGNON operand and KEYMASKED value with the RDEFINE or RALTER command.

Encrypting the Secured Signon Application Key: You can encrypt the secured signon application keys when a common cryptographic architecture (CCA) cryptographic product is installed on the systems where the secured signon function is installed.

Using a cryptographic product ensures the maximum possible security for the secured signon application keys.

With a cryptographic product, RACF can store the keys on the RACF database in a form in which they are encrypted under the cryptographic product’s master key. RACF uses the functions of the cryptographic product to ensure that the encrypted keys do not exist in clear-text form within system main storage for RACF processing, except when they are being defined. Therefore, if a system storage dump occurs, they are not exposed in the dump.

Sharing a RACF Database

- If you want to encrypt the secured signon application keys when a cryptographic product is installed on one or more of the systems that share a RACF database, but is not installed on *all* of the systems, you must ensure that the applications requiring the encrypted keys run *only* on the systems on which the cryptographic product is installed.
- If there is a possibility that an application might run on a system that does *not* have a cryptographic product installed, you must mask the secured signon application keys.

Note: When using the secured signon facilities with encryption, some of the Integrated Cryptographic Service Facility (ICSF) modules must be put in the link pack area (LPA) or the modified link pack area (MLPA) so these modules can be accessed from RACF. The modules that must be put in the LPA or MLPA are:

CSNBCKI

CSNBDEC
CSNBENC
CSNBKRC
CSNBKRD
CSNBKRW

Depending on the release of ICSF, some of these modules may not exist. RACF checks ICSF and only uses existing modules.

To encrypt the secured signon application key when you define or alter it, use the SSIGNON operand and KEYENCRYPTED value with the RDEFINE or RALTER command.

Example of Defining a PTKTDATA Class Profile

Suppose you want to define a profile for TSO in the PTKTDATA class. The system programmer has told you that a VTAM generic resource name for TSO is not being used, and that the SMF identifier of the system on which the TSO application is to run is R001. You want to mask the secured signon application key and specify a key value of X'E001193519561977'. The universal access is to be the default for the PTKTDATA class (NONE).

To define the profile, enter:

```
RDEFINE PTKTDATA TSOR001 SSIGNON(KEYMASKED(E001193519561977))
```

When the Profile Definitions Are Complete

After you define the PTKTDATA-class profile for the application program that is to generate a PassTicket, the program can be installed and used.

For information on how to code an application program to generate a PassTicket, see *z/OS SecureWay Security Server RACF Macros and Interfaces*.

How RACF Processes the Password or PassTicket

To validate a password or PassTicket, RACF does the following:

1. Determines whether the value in the password field is the RACF password for the user ID.
   - If it is the RACF password, the validation is complete.
   - If it is not the RACF password, processing continues.
2. Determines whether a secured signon application profile has been defined for the application in the PTKTDATA class.
   - If a profile has not been defined, the user receives a message from the application[3] indicating that the password is not valid.
   - If the application is defined to the PTKTDATA class, processing continues.
3. Evaluates the value entered in the password field. The evaluation determines whether:
   - The value is a PassTicket consistent with this user ID, application, and time range.
   - It has been used previously on this computer system for this user ID, application, and time range.

   Time Considerations:

3. RACF sends a message to the SYSLOG and to the security console. The application rejects the logon request the same way it rejects an incorrect password. The text of the message the user receives depends on the application.

A PassTicket is considered to be within the valid time range when the time of generation, with respect to the clock on the generating computer, is within plus or minus 10 minutes of the time of evaluation, with respect to the clock on the evaluating computer.

Be sure that your MVS system and the evaluating computer use clock values that are within that time range. RACF uses the value stored for coordinated universal time (UTC), formerly called Greenwich mean time (GMT), in the algorithms that process PassTickets.

One way to ensure that reasonably synchronized values are used is to set UTC in the GMT value of the MVS time of day (TOD) clock and to set a similar value in each of the other systems with which RACF shares PassTicket information. You can still use the MVS local time for local timestamp information, and resetting the local time does not affect the GMT value kept in the TOD clock.

Attention

Before setting the TOD clock's GMT value to UTC, make sure that the subsystems and applications you use are not affected.

To be sure the MVS system clock is set properly, the system console operator should issue:

```
DISPLAY T
```

The system displays the time with information similar to the following:

```
IEE136I LOCAL: TIME=14.06.18 DATE=1997.309
          GMT: TIME=19.06.18 DATE=1997.309
```

Attention

If the MVS `DISPLAY T` command indicates that your system clock is not set correctly for GMT, you need to analyze the consequences of resetting the clock. It is possible that other programs that execute on the system have been adjusted to tolerate an incorrect GMT setting. You may need to readjust those programs before resetting the system clock.

See *z/OS MVS Initialization and Tuning Reference* and *z/OS MVS System Commands* for more information on setting clocks. See *z/OS SecureWay Security Server RACF Macros and Interfaces* for more information on the algorithms.

- If the value was used before, and if PassTicket replay protection has not been bypassed, the user receives a message from the application[4] indicating that the password is not valid.
- If the value was not used before, the PassTicket is considered valid and processing continues.

Determines whether the value is a valid PassTicket.

- If the PassTicket is valid, RACF gives the user access to the desired application.

4. RACF sends a message to the SYSLOG and to the security console. The application rejects the logon request the same way it rejects an incorrect password. The text of the message the user receives depends on the application.

- If the value is not valid, the host application sends a message[5] to the user indicating that the password is not valid.

Note: If the secured signon application key is encrypted, the cryptographic product must be active when RACF tries to authenticate the PassTicket. If it is not active, RACF cannot validate the PassTicket. The resulting message indicates that the logon attempt failed.

Bypassing PassTicket Replay Protection

The option to bypass PassTicket replay protection allows for cases where multiple end-users share the same user ID, and may at times request access to the same application during the same one-second time interval. This option also allows the ″plus or minus 10-minute″ PassTicket replay protection to be bypassed for selected applications or combinations of selected applications, users, or groups.

You indicate that replay protection is to be bypassed for a particular application by adding the text string `NO REPLAY PROTECTION` to the APPLDATA field of the PTKTDATA profile for that application. You must separate each word in the string with a single blank space, alphanumeric character, or keyboard symbol. The `NO REPLAY PROTECTION` text string will always be rolled to upper case by the RALTER or RDEFINE commands.

The `NO REPLAY PROTECTION` text string may appear anywhere within the APPLDATA field, allowing for the existence of other information already in the field, or for new information that may be added in the future.

The following are examples of commands that will cause PassTicket replay protection to be bypassed:

```
RALTER  PTKTDATA profile-name APPLDATA('NO REPLAY PROTECTION')
RDEFINE PTKTDATA profile-name APPLDATA('NO REPLAY PROTECTION')
RDEFINE PTKTDATA profile-name
        APPLDATA('FOR THIS APPLICATION NO REPLAY PROTECTION IS IN EFFECT')
```

Notes:

1. The option to bypass PassTicket replay protection should only be used in secure environments where access to generated PassTickets is limited within a secure or internal network.
2. Other than the APPLDATA (application data) field of the application profile containing the text string, `NO REPLAY PROTECTION`, there is no other external indication that replay protection is bypassed.

Enabling the Use of PassTickets

To enable RACF to validate PassTickets, the RACF administrator must have:

- Activated the PTKTDATA class.
- Defined a secured signon application key for each application in a profile in the PTKTDATA class.
- Issued the SETROPTS RACLIST(PTKTDATA) command.

Note: If you make any additions or changes to profiles in the PTKTDATA class after you issue this command, be sure to re-issue it using the REFRESH option in order to activate your changes.

5. See the previous footnote.

As a result, the RACF database contains all the information necessary to validate PassTickets for each application that has a PTKTDATA class profile defined.

Verifying the Secured Signon Environment

After activating the secured signon environment for each application, you should verify the environment. To do this, access the application using the generated PassTicket from a user ID that is able to access that application. This verifies that:

- The application profile and the secured signon application key have been implemented correctly.
- The secured signon environment is active.

Preventing Errors

The following checklist describes the errors that may cause a PassTicket to fail. To prevent these errors from occurring:

1. Read the list before you use the PassTicket.
2. Review your process to ensure that you have entered all of the information correctly.
3. Verify the information by using the procedures described in “Verifying the Secured Signon Environment”.

Use this checklist to prevent or correct errors:

☐ The PTKTDATA class is activated.

☐ You issued the SETROPTS RACLIST(PTKTDATA) command.

☐ You issued the SETROPTS RACLIST(PTKTDATA) REFRESH command after defining the profile.

☐ A PTKTDATA class profile exists for the application.

☐ You issued the RDEFINE command correctly.

Figure 10. Problem Prevention Checklist

Even if you have followed the proper procedures, it is still possible to receive a message stating that a password is incorrect and be denied access to the application. This can occur if:

- PassTicket replay protection is not being bypassed, and the PassTicket was used previously for this user, application, and time range.

  In this case, RACF generates an SMF record that logs an attempt to replay a PassTicket.
- The GMT clock on the evaluating computer is outside the valid time range for the PassTicket.

  This can be caused by one of the following:
  - The GMT clock on the generating computer and the clock on the evaluating computer are not reasonably synchronized.
  - The PassTicket was not used within approximately ten minutes of being generated.
  - The system clock on the evaluating computer may not be set correctly in relation to GMT. See “Time Considerations” on page 246 for more information.

Protecting the Vector Facility

If your processor has a vector facility, you can use RACF to protect it.

To do this, take the following steps:

1. Define the IEAVECTOR profile in the FACILITY class:

   ```
   RDEFINE FACILITY IEAVECTOR UACC(NONE)
   ```

   This command specifies that *no* users can use the vector facility.
2. Give READ access authority to appropriate users or groups:

   ```
   PERMIT IEAVECTOR CLASS(FACILITY) ID(user or group) ACCESS(READ)
   ```
3. Activate the FACILITY class (if it is not already active):

   ```
   SETROPTS CLASSACT(FACILITY)
   ```

If your installation does not need to control the use of the vector facility, you can define an entry for IEAVECTOR in the global access checking table. Global access checking allows your installation to bypass normal RACF authorization checking and, thereby, minimize processing.

To define an entry for the vector facility in the global access checking table, issue the following commands:

```
RDEFINE  GLOBAL FACILITY
RALTER   GLOBAL FACILITY ADDMEM(IEAVECTOR/READ)
SETROPTS GLOBAL(FACILITY)
```

For more information, see “Setting Up the Global Access Checking Table” on page 206.

Program Control

Program control is a RACF function that allows an installation to treat load modules (programs) as protected resources. This makes the program a “controlled program”. Also, an installation may wish to make the use of a protected program to be a condition for access to a specific data set. This latter function is known as *program access to data sets (PADS)* and is covered in “Program Access to Data Sets (PADS)” on page 252.

The Functions of Program Control

This section describes:

- Protecting load modules as controlled programs
- Protecting program libraries
- Program access to data sets (PADS)

Protecting Load Modules as Controlled Programs

The purpose of protecting load modules is to provide installations with the ability to control *who* can execute *what* programs and to treat those programs as assets.

You can request auditing of attempted accesses to controlled programs (for example, by specifying the AUDIT operand on the RDEFINE or RALTER commands for PROGRAM profiles). You can also specify a NOTIFY user ID for profiles in the PROGRAM class.

You protect individual load modules (programs) by creating a profile for the program in the PROGRAM general resource class. A program protected by a profile in the PROGRAM class is called a *controlled program*.

The name of the profile can be complete, in which case the profile protects only one program, or the name of the profile can end with an asterisk (*), in which case the profile can protect more than one program. For example, a profile named ABC* protects all programs that begin with ABC, unless a more specific profile exists. In this way you can find which programs are causing the environment (such as PADS checking) to not work.

Programs reside in program libraries. A program library is a partitioned data set whose members are load modules. Program libraries can be for public use (those in LNKLIST) or for limited private use. To set up program control, you must protect the library from which the program is loaded. “Protecting Program Libraries” on page 252 discusses the ways to protect the two types of libraries.

The profile for a controlled program *must* also include the name of the program library that contains the program. The profile can also contain:

- The volume serial number of the volume that contains the program library
- A standard access list of users and groups and their associated access authorities
- A conditional access list of users or groups and system ID combinations used to restrict access to a program based on the system identifier (SMFID).

GENERICOWNER is not supported for the PROGRAM class. (All profiles in the PROGRAM class are discrete profiles. Even though profiles ending in * are allowed in this class, these are not true generic profiles). If you want to control creation of profiles in the PROGRAM class, CLAUTH(PROGRAM) may be used.

Defining profiles for load module names in the PROGRAM class does the following:

- Authorizes execution of programs from specific program libraries.
- Controls the execution of programs using alias names. The program name and its alias both need to be defined in the PROGRAM class. TSO commands can be controlled if they are not in the link pack area (LPA).

  The following example controls the use of the DELUSER command and its associated alias DU, and authorizes only the SECADMIN group to use them.

```
RDEFINE PROGRAM DELUSER UACC(NONE) ADDMEM('SYS1.LINKLIB'//NOPADCHK)
RDEFINE PROGRAM DU      UACC(NONE) ADDMEM('SYS1.LINKLIB'//NOPADCHK)

PERMIT DELUSER ID(SECADMIN) ACCESS(READ) CLASS(PROGRAM)
PERMIT DU      ID(SECADMIN) ACCESS(READ) CLASS(PROGRAM)
```

- Controls only load modules that are protected by profiles in the PROGRAM class.

  Note: If program access to data sets (PADS) is being used, a profile in the PROGRAM class must be defined for the load module that actually issues the OPEN macro. In other words, if PADS is in use, the program that opens the data set must be a controlled program.

Access control to load modules *does not:*

- Control the execution of CLISTs or JCL procedures.
- Protect programs from in-memory copying or dumping. See “Controlling Access to Program Dumps” on page 267.

- Control programs or commands that are in the LPA.
- Provide any protection within a multiple-user address space (such as CICS/ESA). If one user loads a program, another user in the same address space can also execute the program.
- Control programs that are executed in any way that bypasses MVS contents supervision (for example, some program invocations by IMS, or programs that reside in z/OS UNIX files). Therefore, loading such a program prevents you from opening a data set in a PADS environment, and prevents you from loading a program from an MVS library if you only have EXECUTE authority. In these cases, installations should use program control to restrict access to any programs that provide facilities for bypassing MVS contents supervision.

Program Control by System Identifier (SMFID)

Program control by system identifier provides a way to restrict access to a program based on system identifier (SMFID). The SMFID is the SID value specified in the SMFPRM*xx* member of the parameter library.

To set up program control by system ID, create a conditional access list for the PROGRAM profile that protects the program. To do this, specify WHEN(SYSID(*system-id*)) with the ID and ACCESS operands on the PERMIT command:

```
PERMIT profile-name CLASS(PROGRAM) ID(user or group) ACCESS(READ)
       WHEN(SYSID(system-identifier))
```

The WHEN(SYSID) operand requires the user to run the program from the specified system.

Protecting Program Libraries

Program libraries can be for public use or for limited private use. An installation designates a set of libraries as public by placing the libraries in the LNKLIST concatenation (which is SYS1.LINKLIB and any program libraries concatenated to SYS1.LINKLIB through the use of the LNKLST*xx* member of SYS1.PARMLIB).

A private load library is a partitioned data set that contains load modules and is not in the LNKLIST. Load modules in private libraries can be protected as execute-controlled libraries. When EXECUTE is the highest access level that users have to a private load library, the library is called an *execute-controlled library*.

To prevent an unauthorized user from copying a program, renaming it to a name that is unknown to program control, and then executing the renamed program, you should protect any program library that contains RACF-controlled programs with data set profiles that have a UACC of NONE or EXECUTE.

Program Access to Data Sets (PADS)

Program access to data sets allows an authorized user or group of users to access specified data sets with the user's authority to execute a certain program. That is, some users can access specified data sets at a specified access level only while executing a certain program (and the program access is limited to controlled programs).

To set up program access to data sets, create a *conditional access list* for the data set profile protecting the data sets. To do this, specify WHEN(PROGRAM(*program-name*)) with the ID and ACCESS operands on the PERMIT command. Specifying the WHEN(PROGRAM) operand requires that the user or group specified must be running the specified program to receive the specified access.

Note: Specifying ALTER in a conditional access list provides no more authority than UPDATE (for non-VSAM data sets) or CONTROL (for VSAM data sets). Specifically, ACCESS(ALTER) with WHEN(PROGRAM(...)) does not allow users to delete the data set.

If PADS is to be used, the following programs *must be specified* on the conditional access list for the data set:

- All programs that are connected to a task's program request block (PRB) at the time of OPEN.
- If a module links to a module that does the OPEN, the program that does the OPEN.
- If loading and calling, the program that does the load.

To create an entry in the conditional access list of a data set profile, issue the PERMIT command:

```
PERMIT 'XXX.YYY' ID(DAVIS) ACCESS(READ) WHEN(PROGRAM(ABC001))
```

The entry you create with this command allows user DAVIS READ access to the data set protected by profile XXX.YYY when executing program ABC001.

Notes:

1. When you are creating an entry in a conditional access list, the program name you specify cannot contain any generic characters. For example, if you specify WHEN(PROGRAM(*)), WHEN(PROGRAM(IKJ*)), or WHEN(PROGRAM(ABC00%), RACF ignores the command. You must specify the program name exactly. For more information, see *z/OS SecureWay Security Server RACF Command Language Reference*.
2. Program access to data sets does not allow different programs with the same name in different libraries with different access requirements.
3. ALTER access authority in a conditional access list does not allow the user to create or delete the data set.
4. If users want to create a particular data set while running a particular program, and if the data set is to be allocated through JCL, do not give these users ALTER authority through PADS. PADS authority checking occurs when the data set is opened, not when it is allocated.

   If users want to control allocation through PADS, they can write an application to use dynamic allocation (SVC99) to allocate a data set rather than JCL. Users can then be allowed to create the data set while running the new program. The new program could then invoke the original application program.

Protecting Programs and Using Them with PADS

Protecting programs and using them with PADS involves:

- Defining and maintaining entries in the PROGRAM class
- Defining protected program libraries
- Defining data sets with conditional access

Defining and Maintaining Entries in the PROGRAM Class

Use the RDEFINE and RALTER commands to define and maintain entries in the PROGRAM general resource class. The load module name is the resource name. The ADDMEM operand describes the library containing the module. When you define the load module name, you can specify an asterisk (*) at the end of the name.

You can specify NOTIFY and auditing for profiles in the PROGRAM class, and for the PROGRAM class itself. However, you cannot request the gathering of statistics.

Choosing between the PADCHK and NOPADCHK Operands: With the ADDMEM operand of the RDEFINE and RALTER commands, you can also specify PADCHK or NOPADCHK.

PADCHK (the default) means that RACF checks for program-accessed data sets that are *already open* before executing the program. For more information about PADS, see "Program Access to Data Sets (PADS)" on page 252. If there are any open program-accessed data sets, RACF ensures, before it allows this program to be loaded, that this user ID/program combination is in the conditional access list of each data set.

If the user-ID/program combination does not have sufficient authority for each data set, the program is not loaded.

NOPADCHK means that RACF does not perform the program-accessed data checks for the program. The program is loaded and has access to any currently opened program-accessed data sets, even though the user ID/program combination is not in the conditional access list. NOPADCHK allows an installation to define entire libraries of modules (such as the PL/I transient routines or ISPF) as controlled programs without having to give each of these modules explicit access to many program-accessed data sets. Use NOPADCHK if you trust the programs to access only data they should.

Note: A user can regain a controlled environment by issuing the TSOEXEC command:

```
TSOEXEC CALL MYPGM
```

When writing a program, you can do the equivalent by invoking the TSO IKJEFTSR service if the service invokes the authorized program or command. For information on using the IKJEFTSR service, see *z/OS TSO/E Programming Services*.

For best results, do not mix PADCHK and NOPADCHK definitions in the same program library. When some members of a program library have been defined with PADCHK and some with NOPADCHK, PADCHK is used for all members.

Defining Protected Program Libraries (Including Execute-Control)

Program libraries that contain RACF-controlled programs must be protected by defining profiles in the DATASET class to cover them.

For public libraries (those in LNKLIST), use ADDSD to define a data set profile that covers the library. The universal access (UACC) for this data set profile can be NONE. Users can still use the controlled programs and any other program in those libraries because the system opens the LNKLIST libraries during IPL and makes the programs public. Users are prevented from opening these libraries and copying the modules in them. The UACC for libraries in the LNKLIST can be set to EXECUTE with the same results as if UACC were NONE. If a LNKLIST library is protected with UACC of EXECUTE, the library is treated as an execute-controlled library (see below for more information) if a user tries to open it (for example through a JOBLIB DD card).

For private libraries, you can establish them as *execute-controlled* libraries by using ADDSD to define a data set profile that covers the library. You must ensure that EXECUTE is the highest access authority given to users who do not need to

maintain the library. For example, you might use the ADDSD or ALTDSD command to specify EXECUTE as the universal access authority (UACC) for a program library, and use the PERMIT command to give UPDATE access to the programmer responsible for maintaining the library. This would allow all users to execute programs from the library, and allow the programmer to maintain the library. EXECUTE access authority prevents users from using a utility such as IEBCOPY to copy a program from the library. However, an authorized user can use MVS system macros (such as LINK, LOAD, XCTL, or ATTACH), the PGM parameter on a JCL EXEC statement, or the TSO CALL command to execute the programs in the library.

All programs in an execute-controlled library must be *controlled programs*. This means that you must define all programs (which are members of the program library) by defining discrete or generic profiles for them in the PROGRAM class. You can then specify universal access authority (UACC) and access lists for the PROGRAM profiles to control which programs in the program library can be run by which users. For example, you might specify a PROGRAM profile named ABC* to protect all programs that begin with ABC.

Although RACF allows programs that are not controlled programs to reside in an execute-controlled library, you may get results that you do not expect and it may be difficult to determine why. This is because when you load a program from an execute-controlled library, RACF tries to preserve the integrity of the program execution environment, as shown by the following example.

When a user who has EXECUTE permission to a library tries to load a program from that library, RACF checks to make sure that the environment is “clean,” that is, that only controlled programs have previously been loaded. If the controlled program calls an uncontrolled program, the uncontrolled program is loaded, but RACF notes that the environment is now “dirty,” that is, that an uncontrolled program has been loaded. If the uncontrolled program in turn calls a controlled program from the execute-controlled library, the controlled program is not loaded, because the environment is no longer clean.

Other sequences are possible, and the outcomes depend on the sequence of calls. Therefore, for best results, control all programs that reside in an execute-controlled library.

User IDs with the RESTRICTED attribute require special consideration because they are prevented from accessing protected resources through the ID(*****) entry on the access list or the UACC. They must be specifically authorized with sufficient authority on the access list of any protected resource they need to access.

Defining Program-Accessed Data Sets with Conditional Access Lists

To create a conditional access list and allow users to access a program-accessed data set, issue the PERMIT command with WHEN(PROGRAM(*program-name*)) specified. Note that it is possible for a user ID to exist in both the standard access list and the conditional access list of a profile of a program-accessed data set. For example, the following commands allow USER01 to read the data set SYS1.LINKLIB anytime, but USER01 can update SYS1.LINKLIB only when running the program PROGA, which is a controlled program:

```
PERMIT 'SYS1.LINKLIB' ID(USER01) ACCESS(READ)
PERMIT 'SYS1.LINKLIB' ID(USER01) ACCESS(UPDATE) WHEN(PROGRAM(PROGA))
```

How Protection Works for Programs and PADS

This section describes:

- How program control works
- Informational error messages for program control
- Authorization checking for access control to load modules
- Authorization checking for program access to data sets
- How RACF and DFP handle execute-controlled libraries

How Program Control Works

The WHEN(PROGRAM) operand on the SETROPTS command activates and deactivates program control. (Note that you need not activate the PROGRAM class to have program control active.) When program control is active, RACF builds, during RACF initialization, an in-storage profile table composed of the entries in the PROGRAM class (controlled programs). The table entries describe the programs and who can access them. To refresh this table, issue SETROPTS WHEN(PROGRAM) REFRESH.

When program control is active, the contents supervision component of MVS invokes RACF, by issuing a RACROUTE REQUEST=FASTAUTH macro, before processing each request to load a module. If the user is not authorized to execute the program, the system issues abend code 306 or 806 and ends the request.

Note: If a non-APF authorized program issues a LINK or LOAD, and passes directory information through the DE operand, and the DE information is for a controlled program, contents supervision reissues the BLDL macro just as it would if the DE information indicated that the requested module was coming from an APF-authorized library.

Informational Messages for Program Control

RACF provides several informational error messages in support of your implementation of program control. These messages are provided by the IRRENS00 environment service for diagnostic purposes, and are particularly helpful for enabling z/OS UNIX servers and daemons in secure systems where the BPX.DAEMON and BPX.SERVER resources are controlled in the FACILITY class. See *z/OS UNIX System Services Planning* for more information about enabling z/OS UNIX servers and daemons.

The IRRENS00 environment service provides informational messages related to the following IRRENS00 functions:

1. Verify and keep the current execution environment controlled or *clean*

   An environment is considered controlled when all executing programs are trusted. This means that each program is protected by a profile in the PROGRAM class. When IRRENS00 is called to verify and keep an environment controlled, it provides informational error messages if the environment is already uncontrolled. Once IRRENS00 has verified that an environment is controlled, it will then keep the environment controlled and provide informational error messages to any user who attempts to make the environment uncontrolled.

2. Mark the current execution environment uncontrolled or *dirty*

   An environment is considered uncontrolled when one or more executing programs are not protected by a profile in the PROGRAM class. When IRRENS00 is called to mark an environment uncontrolled, it provides informational error messages to any user who attempts to execute a controlled program or open a program-accessed data set in the uncontrolled environment.

IRRENS00 also provides helpful informational error messages about authorization failures related to data sets with conditional access lists, and attempts to load uncontrolled modules in the presence of execution-controlled modules.

For examples of error scenarios and informational error messages, see “Examples of Execution-time Errors” on page 264.

Authorization Checking for Access Control to Load Modules

When contents supervision invokes RACF to authorize the loading of a module, RACF makes several checks. Some of these checks involve *program-accessed data sets*. For more information on program-accessed data sets, see “Program Access to Data Sets (PADS)” on page 252.

The checks that RACF makes when a user makes a request to load (execute) a program are:

1. Whether program control has been activated with SETROPTS WHEN(PROGRAM).
2. If program control is active, RACF checks to see whether the program is protected by a profile in the PROGRAM class.
3. If the program is not protected, RACF determines whether there are *any* program-accessed data sets currently open.
   - If there are no program-accessed data sets, RACF causes the system to set an indicator (the TCBNCTL bit in the TCB control block) that a non-controlled program has been loaded, and allows the user to execute the program.
   - If there are currently opened program-accessed data sets, RACF causes the system to end the task with abend code 306 or 806. The user may see message ICH424I specifying one or more currently opened program-accessed data sets. In this way, RACF prevents a controlled program from giving control to a non-controlled program while there are data sets open to which the new user/program combination is not authorized.
4. If the program is protected by a profile but the user does not have at least EXECUTE authority to the program, RACF causes the system to end the task (with abend code 306 or 806) because the user is not authorized to execute the program.
5. If the program is protected by a profile and the user is authorized to execute the program, RACF checks whether any program-accessed data sets are open (unless NOPADCHK was specified for the program).
   - If no program-accessed data sets are open, RACF allows the user to execute the program.
   - If program-accessed data sets are open, RACF checks the user/program combination to verify that the combination has at least the same authority to each data set in the list that was required when each data set was opened. (For more information on the requirements, see “Program Access to Data Sets (PADS)” on page 252.)
     – If the user/program combination has sufficient authority to all of the opened data sets, RACF allows the user to execute the program.
     – If the user/program combination does not have sufficient authority to all of the opened data sets, RACF causes the system to end the task (with abend code 306 or 806).

Note: If you are denied access to a requested resource and you've implemented program control (with or without PADS), see the *z/OS SecureWay Security Server RACF Diagnosis Guide* for help in determining the cause of the authorization failure.

Authorization Checking for Program Access to Data Sets

Whenever a RACROUTE REQUEST=AUTH is issued, RACF performs "normal" authorization checking for access to a data set. That is, RACF grants the request if the UACC is sufficiently high, if the user's user ID is in the access list with sufficient authority, and so forth. If the user is not granted access to the data set with "normal" authorization checking, RACF checks the data set's conditional access list if program control is active and the program currently executing is executing as a RACF-controlled program.

RACF authorizes the user to open the program-accessed data set with the currently executing program if all of the following conditions are met:

- The program name is in the conditional access list of the data set's profile with at least the requested level of authority.
- The user's group or user ID is associated with the program name in the conditional access list.
- The current task did not previously load a non-controlled program, or if any other task in the address space previously loaded a non-controlled program, that task must not be dispatchable. If either of these conditions are not met, the environment is considered uncontrolled. The user's attempt to open the program-accessed data set will fail and the task will end with abend code 913. The user may receive message ICH417I specifying the name of the data set that failed to open due to the uncontrolled environment. For an example of this error, see Figure 11 on page 265.
- If there is more than one controlled program under the dispatchable task, all of those programs (user/program combinations) must be authorized to access the data set. If one or more programs in the environment are not authorized, the attempt will fail and the task will end with abend code 913. The user may receive message ICH418I specifying one or more programs that were missing from the conditional access list. This requirement ensures that no user can execute a program that does not have access to the data set, and then link it to another program that does have access to the data set, and vice versa. For an example of this error, see Figure 12 on page 265.

Note: If a TSO user has executed a non-controlled program during the current session, and then attempts to access a program-accessed data set, the attempt fails. The TSO user can either log off and log back on, or temporarily regain a controlled environment by invoking the controlled program through the TSOEXEC command. When writing a program, you can do the equivalent by invoking the TSO IKJEFTSR service. For information on using the IKJEFTSR service, see *z/OS TSO/E Programming Services*.

How RACF and DFP Process Execute-Controlled Libraries

RACF and DFP process execute-controlled libraries by:

1. Opening an execute-controlled library
2. Fetching a program from an execute-controlled library

Opening an Execute-Controlled Library: When a user issues a request that opens a protected program library, DFP invokes RACF and requests READ access to the library. If EXECUTE is the highest access authority that the user has been given to the library, RACF informs DFP. DFP opens the library and indicates in the user's address space that it is an execute-controlled library.

In order to load a program, the program libraries from which it comes must be opened. The user does not necessarily code the OPEN, for example, a JOBLIB or STEPLIB statement causes an OPEN to occur before the module is fetched.

Notes:

1. Libraries in the LNKLIST concatenation are opened during IPL, and the programs in them are available to anyone unless the program name is defined as a controlled program.
2. If a concatenation containing a data set with UACC(EXECUTE) is opened, the entire concatenation is defined as controlled.

If the user has any other access authority to the library (READ, UPDATE, CONTROL, or ALTER), the library is opened and the user is granted that access authority.

Fetching a Program from an Execute-Controlled Library: After an execute-controlled library is opened, the user can attempt to execute (fetch) a program from the library at any time. RACF checks the user's address space as follows to ensure that it is a controlled environment:

- If the user's address space contains a non-controlled program (a program not defined to RACF in the PROGRAM class), the environment is not controlled. RACF does not allow a program from an execute-controlled library to be fetched into an environment that is not controlled. Therefore, the user's request to load a program from an execute-controlled library will fail, and the task will end with abend code 306 or 806. The user may receive message ICH419I specifying the program that failed to load, and message ICH420I specifying the program that caused the environment to become uncontrolled. For an example of this error, see Figure 13 on page 266.
- If the user's address space contains only controlled programs (or no programs at all), the environment is controlled. RACF allows a program from an execute-controlled library to be fetched into a controlled environment. Thus, RACF grants the user's request to fetch a program from the library.

  However, if the user attempts to fetch a non-controlled program into a controlled environment, RACF ensures that the user has at least READ authority to all RACF-defined programs that are currently residing in the user's address space. RACF does not allow a user to fetch a non-controlled program into an address space that contains a program to which the user has only EXECUTE authority. The user's attempt to load the non-controlled program will fail and the task will end with abend code 306 or 806. The user may receive message ICH423I specifying one or more execute-controlled programs in the current environment. In this way, RACF prevents a user from obtaining a copy of an execute-controlled program by fetching it into a controlled environment and then writing it back to a data set using a non-controlled program. For an example of this error, see Figure 14 on page 266.

If a user submits a request to read a program from an execute-controlled library while in supervisor state, RACF grants the request.

If the user submits any other type of request, such as to edit or copy a program from an execute-controlled library, RACF denies the request.

Note: EXECUTE access authority has meaning only for a partitioned data set that is used as a program library. If you specify EXECUTE for any other type of data set (such as a CLIST or EXEC), RACF treats the access authority as NONE.

Auditing Accesses to Programs in Execute-Controlled Libraries: At the time that an execute-controlled library is opened, OPEN does not know if the user will later attempt to read the library or to execute a program from the library. OPEN issues a RACHECK for READ authority, and RACF fails the request with return code 8 and sets a reason code to indicate that the user did have EXECUTE authority. OPEN examines the reason code and determines that the user has EXECUTE authority. If the user later attempts to read the library, OPEN fails with abend code 913.

At the time when RACHECK detects that the user has only EXECUTE authority, it can not predict if the access will eventually succeed or fail. If the library data set profile indicates that all access attempts should be audited, message ICH408I is issued.

Note: If the library data set profile says to audit both successes and failures, RACHECK interprets this as audit all.

This method of determining access can lead to two confusing scenarios:

1. If the profile says AUDIT(ALL), a user attempting to execute a program can receive message ICH408I saying that she does not have READ authority and then see that the program is successfully executed.
2. If the profile says AUDIT(FAILURES), a user attempting to read the library can get an abend 913 with return code 38 failing the attempt, yet receive no message ICH408I since he did have EXECUTE authority.

In addition, the SMF records produced for an attempt to read and an attempt to execute will be identical for the same reasons described above.

Examples of Controlling Programs and Using PADS

This section contains some examples of:

- Defining load modules as controlled programs
- Setting up program access to data sets (PADS)
- Setting up an execute-controlled library
- Setting up program control by system ID

Examples of Defining Load Modules as Controlled Programs

This section contains examples of:

- Protecting programs without using PADS
- Protecting programs that are in several program libraries
- Protecting all programs on the SYSRES volume using '******'
- Protecting a program on any volume by omitting the volume serial number
- Protecting the AMASPZAP program

Example 1. Protecting Programs Without Using PADS: If you do not intend to use program access to data sets, use the following sequence of commands:

```
1. SETROPTS WHEN(PROGRAM)
       /* activates program control                                    */

2. ADDSD 'SYS1.LINKLIB' UACC(EXECUTE)
       /* prevents users from copying programs                         */

3. RDEFINE PROGRAM MYPROG ADDMEM('SYS1.LINKLIB'/123456/NOPADCHK) UACC(NONE)
       /* makes MYPROG a controlled program. MYPROG must be a member   */
       /* of 'SYS1.LINKLIB' on volume 123456                           */

4. SETROPTS WHEN(PROGRAM) REFRESH
       /* puts the new PROGRAM profile into storage                    */
```

Example 2. Protecting Programs That Are in Several Program Libraries: To protect additional copies of the program in other program libraries, use the ADDMEM operand on the RALTER command:

```
RALTER PROGRAM MYPROG ADDMEM(data-set-name/volser/NOPADCHK)
```

Note that, if the users can STEPLIB to several volumes, you should specify the volume serial numbers for all of the libraries.

Example 3. Using '***' As the Volume Serial Number:*** If you specify **'******'** as the volume serial number, the profile controls programs in all program libraries residing on the SYSRES volume. For example, use this RDEFINE command:

```
RDEFINE PROGRAM program-name ADDMEM(data-set-name/'******'/NOPADCHK)
```

Note: A volume serial number of '******' will work when SYSRES resides on more than one volume, but only applies to the IPL volume.

Example 4. Omitting the Volume Serial Number: You can omit the volume serial number in the ADDMEM, denoting that the controlled program can exist on the specified data set from any volume. For example, use one of these RDEFINE commands:

```
RDEFINE PROGRAM MYPROG ADDMEM(data-set-name//NOPADCHK)
```

or

```
RDEFINE PROGRAM MYPROG ADDMEM(data-set-name)
```

Example 5. Protecting AMASPZAP: The service utility AMASPZAP consists of multiple APF-authorized load modules. Currently, the first module (AMASPZAP) calls a second module (AMAZAP). Most installations want to control the use of the APF-authorized form of this utility.

One approach is to place copies of both load modules in a private library that is not part of LNKLIST but that is designated as APF-authorized in parameter library. Then the installation must link-edit the original AMASPZAP modules again, without APF authorization (AC=0). These original copies can be protected as controlled programs if the installation wishes. To insure that the APF-authorized copies are used only by authorized personnel, the execution of those modules can be protected as execute-controlled programs.

If the AC(0) level load modules have been placed in SYS1.MIGLIB and the function of superzap without APF-authorization will be provided to a limited set of people, protection could be accomplished by the following commands:

```
1. ADDSD 'SYS1.MIGLIB' UACC(EXECUTE) GENERIC
         /* use EXECUTE to limit copying of programs                */

2. RDEFINE PROGRAM AMASPZAP ADDMEM('SYS1.MIGLIB'//PADCHK) UACC(NONE)
         /* this specific profile covers the first module of zap    */
         /* note that any alias names must also be defined          */

   RDEFINE PROGRAM AMAZAP   ADDMEM('SYS1.MIGLIB'//PADCHK) UACC(NONE)
         /* this specific profile covers the second module          */
         /* any future modules added would require specific profiles */

3. PERMIT AMASPZAP CLASS(PROGRAM) ID(group) ACCESS(EXECUTE)
   PERMIT AMAZAP   CLASS(PROGRAM) ID(group) ACCESS(EXECUTE)
         /* gives a limited set of persons access to zap            */

4. SETROPTS WHEN(PROGRAM)
         /* if program control is not already active                */
```

```
   or

   SETROPTS WHEN(PROGRAM) REFRESH
         /* if program control is already active                    */
         /* puts the new PROGRAM profile into storage               */
```

Assuming that the AC(1) level load modules are in a library named SYS1.PRIVLIB on volume D80LIB and SYS1.PRIVLIB is defined as an APF-authorized library, protection could be accomplished by the following commands:

```
1. ADDSD 'SYS1.PRIVLIB' UACC(EXECUTE) GENERIC
         /* use EXECUTE to limit copying of programs                */

2. RDEFINE PROGRAM AMA* ADDMEM('SYS1.PRIVLIB'/D80LIB/PADCHK) UACC(NONE)
         /* this profile covers both of the modules of superzap     */
         /* However, future modules are not guaranteed to be named  */
         /* according to the same convention                        */

3. PERMIT AMA* CLASS(PROGRAM) ID(SYSTEMS) ACCESS(EXECUTE)
         /* gives the SYSTEMS group access to zap                   */
         /* however, they have to STEPLIB to PRIVLIB                */

4. SETROPTS WHEN(PROGRAM)
         /* if program control is not already active                */

   or

   SETROPTS WHEN(PROGRAM) REFRESH
        /* if program control is already active                     */
        /* puts the new PROGRAM profile into storage                */
```

Example of Setting up Program Access to Data Sets

The next section shows an example of protecting the DSPRINT request queue with PADS.

Example of Protecting the DSPRINT Request Queue with PADS: DSPRINT is an unsupported field-developed product of IBM.

Note: Module AAOEFTB3 is part of MVS/TSO Dynamic Steplib Facility, Program Number 5798-DZW. It is not part of DSPRINT.

If you have DSPRINT installed, you should consider protecting the DSPRINT request queue, because it is a sensitive data set that users must not access unless they are running DSPRINT.

Issue the following commands:

```
ADDSD   'DSPRINT.REQUEST.QUEUE' GENERIC UACC(READ)

RDEFINE PROGRAM DSPRINT UACC(READ)

RALTER  PROGRAM DSPRINT ADDMEM('SYS1.PPLINK'/SYSLIB/PADCHK)

RDEFINE PROGRAM AAOEFTB3 UACC(READ)

RALTER  PROGRAM AAOEFTB3 -
        ADDMEM('SYS1.LINKLIB'/IACGR2/PADCHK, -
               'SYS1.LINKLIB'/IAC7R2/PADCHK, -
               'SYS1.LINKLIB'/IAC7R1/PADCHK, -
               'SYS1.LINKLIB'/IACGR1/PADCHK)

PERMIT  'DSPRINT.REQUEST.QUEUE' GENERIC ID(*) -
        ACCESS(UPDATE) WHEN(PROGRAM(DSPRINT))
```

```
PERMIT  'DSPRINT.REQUEST.QUEUE' GENERIC ID(*) -
        ACCESS(UPDATE) WHEN(PROGRAM(AAOEFTB3))

SETROPTS WHEN(PROGRAM) REFRESH
```

When the queue is protected this way, users can execute the DSPRINT program through TSOEXEC.

Example of Setting Up an Execute-Controlled Library

The following sequence of RACF commands illustrates one way you can set up an execute-controlled library. Assume the program is member XCLPGM in program library KBROWN.PGMLIB2. For execute control, you must know the volume on which the program library resides. In this case, library KBROWN.PGMLIB2 is on volume VOL6A.

1. If program control is not active, enter:

   ```
   SETROPTS WHEN(PROGRAM)
   ```

   After program control is active, it remains active until your installation deactivates it by issuing the SETROPTS command with the NOWHEN(PROGRAM) operand.
2. Define a data set profile to protect the private program library by issuing the ADDSD command with the appropriate operands. The following command defines a data set profile to protect program library KBROWN.PGMLIB2. The command assigns a UACC of EXECUTE to allow all users to execute but not otherwise access the library.

   ```
   ADDSD 'KBROWN.PGMLIB2' UACC(EXECUTE)
   ```
3. Define a generic or discrete profile in the PROGRAM class that protects the controlled program. You must specify the ADDMEM operand as shown in the following examples. The following command identifies only program XCLPGM as a controlled program:

   ```
   RDEFINE PROGRAM XCLPGM ADDMEM('KBROWN.PGMLIB2'/VOL6A/NOPADCHK)
   ```

   The following command specifies that all members in KBROWN.PGMLIB2 are controlled programs:

   ```
   RDEFINE PROGRAM ** ADDMEM('KBROWN.PGMLIB2'/VOL6A/NOPADCHK)
   ```

   On the ADDMEM operand, you must specify the following:

   - The name of the program library that contains the program.

     Single quotation marks around the library name are required. Generic characters are not allowed.
   - Whether to check for the program name on the conditional access list of all currently open data sets (PADCHK requires the check; NOPADCHK does not require the check)

   Optionally, you can also include the specific volume on which the program library resides. The string '******' denotes the IPL volume. If no volume is specified, the program library can exist on any volume.

   In these examples, VOL6A is the volume on which KBROWN.PGMLIB2 resides, and NOPADCHK is specified because the programs are trusted to open only data sets that they should.

   Notes:

   a. Execute-controlled programs should not be placed in data sets that are eligible for caching with run-time library services (RTLS). Instead, you'll need

to define a separate physical RTLS library that contains the execute-controlled library and set the cache maximum to 0.

b. If you use ** (instead of *) as the “top” generic profile in the PROGRAM class, you can use the RLIST command to list just the one profile.

c. You can specify ** as the name of only one profile in the PROGRAM class. That one profile can protect several program libraries, as specified on the ADDMEM operand, but they are all protected in the same way (that is, they all have the same UACC, access list, PADCHK or NOPADCHK value, and so forth). You can specify partially-qualified generic names such as ABC* to protect all of the programs that have a similar naming scheme.

4. After defining or changing a profile in the PROGRAM class, refresh the in-storage program control tables by issuing the following command:

```
SETROPTS WHEN(PROGRAM) REFRESH
```

This ensures that the changes take effect immediately.

Example of Setting Up Program Control by System ID

Suppose your installation has two systems in a sysplex and you want to let user Allen run program MYPROG from SYS1 but not from SYS2. You would use these commands:

```
1. SETROPTS WHEN(PROGRAM)
      /* activates program control                                  */

2. ADDSD 'SYS1.LINKLIB' UACC(EXECUTE)
      /* prevents users from copying programs                       */

3. RDEFINE PROGRAM MYPROG ADDMEM('SYS1.LINKLIB'/123456/NOPADCHK) UACC(NONE)
      /* makes MYPROG a controlled program. MYPROG must             */
      /* be a member of 'SYS1.LINKLIB' on volume 123456             */

4. PERMIT MYPROG CLASS(PROGRAM) ID(ALLEN) ACCESS(READ) WHEN(SYSID(SYS1))
      /* user ALLEN can only run the program from system SYS1       */

5. SETROPTS WHEN(PROGRAM) REFRESH
      /* puts the new PROGRAM profile into storage                  */
```

Examples of Execution-time Errors

Errors related to Program-Accessed Data Sets (PADS)

RACF protects program-accessed data sets from being opened by users who are not authorized by the data set’s standard access list, and by user/program combinations that are not authorized by the data set’s conditional access list. Therefore, RACF prevents program-accessed data sets from being opened in uncontrolled environments, and controlled environments where one or more executing programs are not authorized by the conditional access list.

Failure to Open a Program-Accessed Data Set in an Uncontrolled Environment: A user who attempts to open a program-controlled data set in an uncontrolled environment may receive error messages similar to the ones shown in Figure 11 on page 265.

```
ICH417I THE ENVIRONMENT IS NOT CONTROLLED. CONDITIONAL ACCESS
        LIST BYPASSED FOR DATA SET RACFTEST.PADS.DS1
ICH420I PROGRAM PADSOPN1 CAUSED THE ENVIRONMENT TO BECOME
        UNCONTROLLED.
IEA995I SYMPTOM DUMP OUTPUT 389
 SYSTEM COMPLETION CODE=913  REASON CODE=00000038
```

Figure 11. Example of Failure to Open a Program-Accessed Data Set in an Uncontrolled Environment

In Figure 11, informational message ICH417I indicates that data set RACFTEST.PADS.DS1 failed to open because the environment is uncontrolled. Message ICH420I indicates that a currently executing program PADSOPN1 caused the environment to become uncontrolled.

In this case, the user should try to access the program-accessed data set in a controlled environment that does not include the uncontrolled program. If that is not possible, you may consider trusting the uncontrolled program by protecting it with a profile in the PROGRAM class.

Failure to Open a Program-Accessed Data Set in a Controlled Environment: A user who attempts to open a program-controlled data set in a controlled environment where one or more dispatchable programs are not authorized by the data set's conditional access list may receive error messages similar to the ones shown in Figure 12.

```
ICH408I USER(RRSFU3  ) GROUP(SYS1  ) NAME(RRSFU3  ) 562
        RACFTEST.PADS.DS1 CL(DATASET ) VOL(TEMP01)
        INSUFFICIENT ACCESS AUTHORITY
        ACCESS INTENT(READ )  ACCESS ALLOWED(NONE )
IEC150I 913-38,IFG0194E,IBMTESTS,,DD01,0230,TEMP01,
        RACFTEST.PADS.DS1
IEA995I SYMPTOM DUMP OUTPUT 564
 SYSTEM COMPLETION CODE=913  REASON CODE=00000038
ICH418I CONDITIONAL ACCESS LIST FOR DATA SET RACFTEST.PADS.DS1
        DID NOT GRANT AUTHORITY TO PROGRAM(S): PADSPRG1
```

Figure 12. Example of Failure to Open a Program-Accessed Data Set in a Controlled Environment

In Figure 12, informational message ICH408I indicates that user RRSFU3 had insufficient authority to open data set RACFTEST.PADS.DS1 based on the standard access list. Message ICH418I indicates that the currently executing program PADSPRG1 was not found on the conditional access list.

In this case, you may consider adding the user and program to the conditional access list for this data set.

Errors related to Execution-Controlled Libraries

RACF protects execute-controlled programs by not allowing them to be copied by users who are only authorized to execute them. Users who are only authorized to execute an execute-controlled program are prevented from loading it and then using an uncontrolled program to copy it back to a data set. RACF does this by preventing users from loading execute-controlled programs into uncontrolled environments, and by preventing users from loading uncontrolled programs into controlled environments where execute-controlled program are present.

Failure to Load an Execute-Controlled Program into an Uncontrolled Environment: A user who attempts to load an execute-controlled program into an uncontrolled environment may receive error messages similar to the ones shown in Figure 13.

```
CSV025I PROGRAM CONTROLLED MODULE PADS1P01 NOT ACCESSED, USER UNAUTHORIZED
CSV028I ABEND306-30  JOBNAME=IBMTESTS  STEPNAME=
ICH419I THE ENVIRONMENT IS NOT CONTROLLED. ATTEMPT TO LOAD
        PROGRAM PADS1P01 FROM LIBRARY RACFTEST.PADS.LINKLIB2 FAILED.
ICH420I PROGRAM PADSPRG1 FROM LIBRARY RACFTEST.PADS.LINKLIB
        CAUSED THE ENVIRONMENT TO BECOME UNCONTROLLED.
IEA995I SYMPTOM DUMP OUTPUT 651
 SYSTEM COMPLETION CODE=306  REASON CODE=00000030
```

Figure 13. Example of Failure to Load a Controlled Program into an Uncontrolled Environment

In Figure 13, informational message ICH419I indicates that the attempt to load program PADS1P01 failed because the environment was not controlled. Message ICH420I specifies that the environment became uncontrolled when program PADSPRG1 was previously loaded.

In this case, the user should try to execute the execute-controlled program in a controlled environment that does not include the uncontrolled program. If that is not possible, you may consider trusting the uncontrolled program by protecting it with a profile in the PROGRAM class.

Failure to Load an Uncontrolled Program into an Controlled Environment Containing Execute-Only Programs: A user who attempts to load an uncontrolled program into a controlled environment that contains one or more programs to which the user has only EXECUTE authority may receive error messages similar to the ones shown in Figure 14.

```
CSV025I PROGRAM CONTROLLED MODULE PADS1P02 NOT ACCESSED, USER UNAUTHORIZED
CSV028I ABEND306-30  JOBNAME=IBMTESTS  STEPNAME=STEPSL
ICH423I RACF EXECUTE-CONTROLLED PROGRAMS ARE ACTIVE: PADSPRG1
IEA995I SYMPTOM DUMP OUTPUT 765
 SYSTEM COMPLETION CODE=306  REASON CODE=00000030
```

Figure 14. Example of Failure to Load an Uncontrolled Program into an Controlled Environment with Execute-Only Programs

In Figure 14, informational message CSV025I indicates that the attempt to load program PADS1P02 failed. Informational message ICH423I indicates that the environment contained the execute-controlled program PADSPRG1.

In this case, you should determine if the user should be authorized to perform this action in the current environment. If so, you can authorize the user to READ the controlled program by changing the access list of the DATASET profile protecting the execute-controlled library, or you may consider trusting the uncontrolled program by protecting it with a profile in the PROGRAM class.

Controlling Access to Program Dumps

Because program dumps can contain sensitive information including controlled programs in machine form, you should consider limiting access to them. To control access to program dumps or suppress the dumps entirely, you can use RACF, operating system facilities (such as SYS1.PARMLIB), JES2, or JES3.

Using RACF to Control Access to Program Dumps

RACF allows you to control access to program dumps selectively. To achieve this control, you can protect program dumps using either a data set profile or a resource profile in the FACILITY class for one of the following resources: IEAABD.DMPAUTH or IEAABD.DMPAKEY.

Protecting Program Dumps Using a Data Set Profile

To protect program dumps using a data set profile:

1. Create a generic profile to protect all dump data sets with a high-level qualifier of SYS1, specifying a UACC of NONE. Enter:

   ```
   ADDSD 'SYS1.DUMP%%' UACC(NONE)
   ```

2. Permit selected users to access the data sets protected by the SYS1.DUMP%% profile by adding them to the access list with READ access authority. Enter:

   ```
   PERMIT 'SYS1.DUMP%%' ID(userid) ACCESS(READ)
   ```

Protecting Program Dumps Using the FACILITY Class

Your installation can control the dumping (with SYSUDUMP, SYSABEND, and SYSMDUMP statements) of address spaces that contain controlled programs by defining a profile to protect a resource called IEAABD.DMPAUTH in the FACILITY general resource class.

To control the dumping (with SYSABEND, SYSMDUMP, and SYSUDUMP statements) of address spaces that have tasks running in a task control block (TCB) key of less than 8, a profile protecting a resource called IEAABD.DMPAKEY must be defined in the FACILITY general resource class. IBM recommends that the profile be defined with UACC(NONE). Then, you can give ACCESS(READ) to specific users using the PERMIT command.

Example of Defining the IEAABD.DMPAUTH Profile:

1. Define a profile protecting a resource called IEAABD.DMPAUTH in the FACILITY class:

   ```
   RDEFINE FACILITY IEAABD.DMPAUTH UACC(READ)
   ```

2. If you want to give a user an access authority that is different from the one you specified on the RDEFINE command (in this example, an access authority of NONE), enter:

   ```
   PERMIT IEAABD.DMPAUTH CLASS(FACILITY) ID(ASMITH) ACCESS(NONE)
   ```

   When you specify an access authority on either the RDEFINE command or PERMIT command, RACF allows access to program dumps as follows:

   - A user who has UPDATE or greater authority to the IEAABD.DMPAUTH resource can always obtain program dumps.
   - A user who has READ authority to the IEAABD.DMPAUTH resource can obtain program dumps unless a program has been fetched from a library to which the user has only EXECUTE authority. The user cannot obtain a dump of a program to which he or she has only EXECUTE authority.

     See “How RACF and DFP Process Execute-Controlled Libraries” on page 258 for more information.

- A user who has less than READ authority to the IEAABD.DMPAUTH resource can never obtain program dumps of address spaces that contain controlled programs.

Example of Defining the IEAABD.DMPAKEY Profile:

1. Define a profile protecting a resource called IEAABD.DMPAKEY in the FACILITY class:

   ```
   RDEFINE FACILITY IEAABD.DMPAKEY UACC(NONE)
   ```

2. If you want to give a user an access authority that is different from the one you specified on the RDEFINE command (in this example, an access authority of READ), enter:

   ```
   PERMIT IEAABD.DMPAKEY CLASS(FACILITY) ID(ASMITH) ACCESS(READ)
   ```

   When you specify an access authority on either the RDEFINE command or PERMIT command, RACF allows access to program dumps as follows:

   - A user who has READ or greater authority to the IEAABD.DMPAKEY resource can obtain program dumps, even when the program is running in a TCB key that is less than 8.
   - A user who has less than READ authority to the IEAABD.DMPAKEY resource can never obtain program dumps when the program is running in a TCB key that is less than 8.

Activating the FACILITY Class:

1. If the FACILITY class is not active, activate it as follows:

   ```
   SETROPTS CLASSACT(FACILITY)
   ```

   You only need to issue this command once. When a general resource class is active, it remains active until your installation deactivates it.

2. To avoid possible deadlocks, issue a SETROPTS RACLIST command for the FACILITY class.

   Example of a Deadlock:

   There are several types of deadlocks. This example describes one way a deadlock can occur.

   - Task A of a job is abending.
     - z/OS needs to check the user's authority to produce a dump of the task and issues RACROUTE REQUEST=AUTH.
     - RACF needs to do I/O to the RACF database to respond to the RACROUTE request.
   - Task B of the same job is already performing a RACF function and has ENQed the RACF database.
   - Task A must wait until task B releases the ENQ.
   - Dump processing for task A has set all other tasks in the job non-dispatchable.

   Under normal circumstances, task A could wait for task B to release the ENQ. However, because dump processing for the abending task prevents task B from completing, task A cannot proceed. Task B cannot complete until task A proceeds. This causes a deadlock.

Using Non-RACF Methods to Control Access to Program Dumps

You can control access to program dumps using a variety of non-RACF methods, whose documentation is beyond the scope of this book. For more information about suppressing and redirecting dump output, see *z/OS MVS Recovery and Reconfiguration Guide*.

Controlling the Allocation of Devices

You can use the DEVICES class to control which users can allocate unit record devices, teleprocessing or communications devices, and graphics devices. For example, you can use the DEVICES class to ensure that only authorized users can allocate devices by name. You cannot use the DEVICES class to protect other kinds of devices, such as tape or DASD devices.

Note: To control who can log on to terminals, see “Protecting Terminals” on page 235. To control who can log on to consoles, see “Protecting Consoles” on page 238.

To control the allocation of devices, do the following:

1. Plan which devices to protect. You can, for example, protect specific devices with discrete profiles. You can also protect several devices with generic profiles.
2. Ask your MVS system programmer to supply the following information for each device to be protected:
   - The information that is necessary to specify the name of the profile that is to protect the device, such as the device class, unit name, and device number. These terms are described in more detail in step 3.
   - The RACF-defined users that can allocate the device that is protected by the profile.

   Note: With this information, you can plan whether to use generic profiles, discrete profiles, or a combination, to protect the devices on your system.

   If you decide to use generic profiles, you must activate generic processing for this class before you define the profiles.

   ```
   SETROPTS GENERIC(DEVICES)
   ```
3. Create profiles in the DEVICES class:

   ```
   RDEFINE DEVICES profile-name UACC(NONE)
   ```

   where *profile-name* has the following format:

   ```
   sysid.device-class.unit-name.device-number
   ```

   where:

   sysid is the system identifier, which is defined on the SYSNAME value in the IEASYS*xx* member of SYS1.PARMLIB.

   Note: The system identifier is necessary only if different devices with the same device class, unit name, and device address can be attached to multiple systems and have different security requirements. In most cases, you should specify an asterisk (*) for this qualifier.

   device-class is one of the following UCB device classes:

   TP Teleprocessing or communication devices

UR Unit record devices

GRAPHIC Graphic devices

Note: These device classes are consistent with the class names used on the DISPLAY U operator command.

unit-name is an esoteric device group (as defined by the installation) or a generic name (such as 3800) that identifies the device or devices.

Note: Because a user can allocate a device using either an esoteric or generic name, you must define profiles that would protect the device in either case.

For telecommunication devices, use the following list to determine what unit name should be used. The unit name that MVS uses for these devices is based on the transmission control unit (TCU) value of the IODEVICE statement.

TCU Value	Unit Name
TCU=2701	2701
TCU=2702	2702
TCU=2703	2703

For all other devices, see *z/OS HCD Planning* for unit name information.

Note: For any device for which MVS does not have a unit name, MVS uses 8 character zeros (for example, 00000000). Use this as the unit name in the profile name to provide security for these devices.

device-number is a 4-byte field that supplies the number of a specific device.

For information about I/O device numbers, see *z/OS HCD Planning*.

Note: If *unit-name* identifies an esoteric device group, specify an asterisk (*) in this qualifier.

Here are some sample profile names for the DEVICES class:

```
SYS01.GRAPHIC.3277-2.B40
SYS01.TP.3705.3FA
SYS01.UR.3800.00E
SYS01.UR.PRINTER1.*
```

Specifying UACC(NONE) means that only users who are specifically permitted to the profile can allocate the device.

4. Give users the appropriate access authority:

```
PERMIT profile-name CLASS(DEVICES)
       ID(userid or groupname) ACCESS(access-authority)
```

where *access-authority* is one of the following:

NONE Does not allow the allocation of the device

READ Allows the allocation of the device.

5. When you are ready to start using the security provided by these profiles, activate both the DEVICES class and SETROPTS RACLIST processing for the class. SETROPTS RACLIST processing helps ensure high performance when access authorities are checked. You can do these two actions in one command:

   ```
   SETROPTS CLASSACT(DEVICES) RACLIST(DEVICES)
   ```

   Note: Any time you make a change to a DEVICES profile, you must also refresh SETROPTS RACLIST processing for the DEVICES class for the change to take effect. For example:

   ```
   SETROPTS RACLIST(DEVICES) REFRESH
   ```

Example 1:

The following commands allow only USER1 to allocate PRINTER1:

```
RDEFINE DEVICES SYS01.UR.PRINTER1.* UACC(NONE)
RDEFINE DEVICES SYS01.UR.3800.00E   UACC(NONE)
PERMIT  SYS01.UR.PRINTER1.* CLASS(DEVICES) ID(USER1) ACCESS(READ)
PERMIT  SYS01.UR.3800.00E   CLASS(DEVICES) ID(USER1) ACCESS(READ)
```

Example 2:

The following commands allow only group DESIGN1 to allocate graphics devices:

```
RDEFINE DEVICES SYS01.GRAPHIC.** UACC(NONE)
PERMIT  SYS01.GRAPHIC.** CLASS(DEVICES) ID(DESIGN1) ACCESS(READ)
```

Protecting LLA-Managed Data Sets

You can control which users can issue the START LLA and MODIFY LLA commands. When a user issues the START LLA and MODIFY LLA commands, the library lookaside facility (LLA) invokes a RACF authorization check. This is done for each parameter library data set that LLA needs to access, and for each LLA-managed data set.

To do this, take the following steps:

1. If data set profiles for each LLA parameter library data set do not currently exist, create them. These parameter library data sets are those containing CSVLLA*xx* members that specify which libraries LLA is to manage and how it is to manage them. Make sure each LLA command user (or a group to which the user belongs) has READ access to all parameter library data sets that you protect.
2. Create profiles in the FACILITY class to protect the LLA-managed data sets. These data sets are the libraries that are specified in the CSVLLA*xx* and LNKLST*xx* members of a parameter library. For example:

   ```
   RDEFINE FACILITY CSVLLA.data-set-name UACC(NONE)
   ```

   where *data-set-name* is the name of the LLA-managed data set.

   Because of the CSVLLA prefix used on the resource names, and because the FACILITY class profiles can only be 39 characters long, the *data-set-name* portion of this profile is limited to 32 characters. If your data set name is longer than 32 characters, use generics so that the FACILITY class profile stays within the 39-character limit.

   Notes:

   a. You should consider creating the same FACILITY profiles as you did data set profiles in step 1.

b. To have this protection, you must create profiles in the FACILITY class as well as the DATASET class if you do not have access to the data set already.

c. The LLA facility first checks the user's access through the FACILITY class profile and, unless this access is allowed, then checks for access through a data set profile.

3. Give users and groups the appropriate access authority:

```
PERMIT CSVLLA.data-set-name CLASS(FACILITY)
       ID(userid or groupname) ACCESS(access-authority)
```

This PERMIT command allows users or groups to issue LLA commands for the specified LLA-managed library. This access authority (*access-authority*) can be one of the following:

NONE Allows no access.

UPDATE Allows users to work with the data sets using the LLA START and LLA MODIFY commands.

ALTER For discrete profiles, allows same access as UPDATE, plus the ability to change the profile itself. For generic profiles, equivalent to UPDATE.

4. If you have not already done so, activate the FACILITY class:

```
SETROPTS CLASSACT(FACILITY)
```

Example:

For example, to control all LLA-managed data sets whose high-level qualifier is CICS, enter:

```
ADDSD   'CICS.*' UACC(NONE)
PERMIT  'CICS.*' ID(CICS) ACCESS(READ)
RDEFINE  FACILITY CSVLLA.CICS.* UACC(NONE)
PERMIT   CSVLLA.CICS.* CLASS(FACILITY) ID(CICS) ACCESS(UPDATE)
SETROPTS CLASSACT(FACILITY)
```

This command sequence allows CICS to issue the LLA MODIFY command for the LLA-managed data sets whose high-level qualifier is CICS.

Controlling Data Lookaside Facility (DLF) Objects (Hiperbatch)

You can use profiles in the DLFCLASS class to control whether data in QSAM or VSAM data sets can be stored in a data lookaside facility (DLF) object and managed by Hiperbatch, an extension to QSAM and VSAM. Hiperbatch can improve the performance of batch jobs by minimizing I/O to the DASD device on which the data sets are stored. For more information on DLF objects, see *MVS Hiperbatch Guide*.

Profiles in the DLFCLASS general resource class provide the following control information to DLF:

- The *existence* of a DLFCLASS profile for a QSAM or VSAM data set identifies the data set as one that is eligible to be processed as a DLF object.
- The RETAIN field in the DLFDATA segment of the profile allows you to indicate to DLF whether the object is to be a retained DLF object. The RETAIN field is an optional field.
- The access list for the profile identifies the users and groups who are allowed to access the DLF object.

- The JOBNAMES field in the DLFDATA segment of the profile allows you to further restrict access to the DLF object to specific job names:
  - When you specify a list of job names, access to the DLF object is allowed only when the user ID appears in the access list and the specific job name appears in the job names list.
  - When you do not specify a list of job names, all jobs submitted by authorized users (that is, users who are identified in the access list of the profile) can access the DLF object.

  The JOBNAMES field is an optional field.

If you do not include this information in DLFCLASS profiles, your DLF installation exit must identify the eligible data sets and retained DLF objects, and also verify user or job name access to a DLF object.

For example, for a QSAM or VSAM data set named PAYROLL.SALARY.DATA, a DLFCLASS profile named PAYROLL.SALARY.DATA allows data from the data set to be stored in a DLF object that is used by jobs accessing the data set.

To use RACF to support DLF objects, take the following steps:

1. Create a discrete profile in the DLFCLASS class:

   ```
   RDEFINE DLFCLASS profile-name UACC(...) DLFDATA(...)
   ```

   where *profile-name* is the fully-qualified data set name (do not specify quotes). The profile must be a discrete profile. For example, for a data set named PAYROLL.SALARY.DATA, the profile name would be:

   ```
   PAYROLL.SALARY.DATA
   ```

   UACC and DLFDATA are optional. Possible options to support a DLF object are:

   UACC Controls access to the DLF object. Access authorities are as follows:

NONE	Allows no access to the DLF object.
READ	Allows the job to read from the DLF object.
UPDATE	Is equivalent to READ
CONTROL	Is equivalent to READ
ALTER	Allows READ access, and also allows users to change the profile (if it is a discrete profile).

   DLFDATA

   - If the DLF object is to be retained, specify RETAIN(YES) in the DLFDATA operand.

     Note: If you do not specify DLFDATA, or if you do not specify the RETAIN value in the DLFDATA operand, RETAIN(NO) is defaulted.
   - If access to the DLF object is to be limited to specific jobs, include the JOBNAMES value in the DLFDATA operand and list all of the applicable job names, as:

     ```
     DLFDATA(JOBNAMES(jobname1 ...))
     ```

   See *z/OS SecureWay Security Server RACF Command Language Reference* for more details and other options.

2. Give users and groups the appropriate access authority. For example:

   ```
   PERMIT profile-name CLASS(DLFCLASS) ID(userid or groupname)
          ACCESS(access-authority)
   ```

3. When you are ready to start using the protection defined in the profiles, activate the DLFCLASS class as follows:

   ```
   SETROPTS CLASSACT(DLFCLASS)
   ```

4. For enhanced performance related to the DLFCLASS profiles themselves, activate SETROPTS RACLIST processing as follows:

   ```
   SETROPTS RACLIST(DLFCLASS)
   ```

Example 1: Limiting Access by Job Name

Users AHLEE and SMITH can both access a DLF object that corresponds to a data set named SALES.DATA, but they can do this only by submitting jobs whose job names begin with TAX, or jobs named TOTALS.

Create a profile for the DLF object:

```
RDEFINE DLFCLASS SALES.DATA DLFDATA(JOBNAMES(TOTALS TAX*)) UACC(NONE)
```

Give users and groups the appropriate access authority.

```
PERMIT SALES.DATA CLASS(DLFCLASS) ID(AHLEE SMITH) ACCESS(READ)
```

If the DLFCLASS class is not active:

```
SETROPTS CLASSACT(DLFCLASS)
```

Example 2: Not Retaining a DLF Object

An interactive application is invoked many times during the day by many users. This application makes many I/O operations to data set PAY.DATA. At the end of the day, the application ends and the need for the DLF object ends. To improve performance when the application is in use, create a DLFCLASS profile for the data set with RETAIN(NO) specified.

Create a profile for the DLF object:

```
RDEFINE DLFCLASS PAY.DATA DLFDATA(RETAIN(NO)) UACC(NONE)
```

Give appropriate access:

```
PERMIT PAY.DATA CLASS(DLFCLASS) ID(PAYGRP) ACCESS(READ)
```

If the DLFCLASS class is not active:

```
SETROPTS CLASSACT(DLFCLASS)
```

Using RACROUTE REQUEST=LIST,GLOBAL=YES Support

The RACROUTE REQUEST=LIST,GLOBAL=YES option creates in-storage profiles in a data space rather than in private storage. It allows multiple address spaces to share the same set of RACLISTed profiles. Each additional application that issues RACROUTE REQUEST=LIST,GLOBAL=YES for the same class can access the data space already built. Profiles that are globally RACLISTed for a class with RACROUTE REQUEST=LIST,GLOBAL=YES can be refreshed simultaneously for all users by SETROPTS RACLIST(*classname*) REFRESH. The refresh occurs without requiring the applications to suspend work.

Classes that are RACLISTed solely by this means are listed in a line in the output of the SETR LIST command, with the following format:

```
GLOBAL=YES RACLIST ONLY =
```

After all applications have given up their access to the RACLIST data space by issuing RACROUTE REQUEST=LIST,ENVIR=DELETE, you can delete the data space by issuing SETROPTS NORACLIST(*classname*). Remember that the SETROPTS RACLIST REFRESH and SETROPTS NORACLIST commands process both the class specified by the command and all valid classes sharing the same POSIT value as the specified class. Additionally, if the system is enabled for sysplex communication, the command is propagated to the other members of the sysplex data sharing group.

For a detailed comparison of the RACROUTE and SETROPTS processes, see *z/OS SecureWay Security Server RACF Diagnosis Guide*.

The RACGLIST Class

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

- SETROPTS RACLIST
- SETROPTS RACLIST REFRESH
- RACROUTE REQUEST=LIST,GLOBAL=YES

RACF uses the results from the data space to create or replace profiles named *classname_nnnnn* (where *nnnnn* begins at 00001) in the RACGLIST class. To enable RACF to use the RACGLIST profiles, the installation must to activate the RACGLIST class and prime RACGLIST for a specific class by issuing the RDEFINE RACGLIST *classname* command, where *classname* is a valid class name. For example, if the RACGLIST class name profile is TCICSTRN, RACF creates additional profiles named TCICSTRN_00001, TCICSTRN_00002, and so forth.

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

- For a subsequent RACROUTE REQUEST=LIST,GLOBAL=YES request on a different system sharing the RACF database
- For SETROPTS RACLIST processing during a system IPL
- After a system IPL by RACROUTE REQUEST=LIST,GLOBAL=YES processing
- During the processing of a propagated SETROPTS RACLIST or SETROPTS RACLIST *classname* REFRESH command

The RACGLIST class provides a single-system image for security when you use RACROUTE REQUEST=LIST,GLOBAL=YES or SETROPTS RACLIST on multiple systems that are enabled for sysplex communication. All systems and regions using a class whose RACLIST results have been saved as *classname_nnnnn* profiles use the same data to make security decisions. Any system that performs a RACROUTE REQUEST=LIST,GLOBAL=YES retrieves the same profiles. When several changes are required for profiles in that class, other systems continue to access the stored profiles until the administrator completes the changes and tells RACF to refresh the profiles by issuing SETROPTS RACLIST(*classname*) REFRESH.

Special Considerations

1. You should use the RACGLIST class only if all systems sharing the RACF database belong to the same global resource serialization (GRS) complex. See the appropriate level publication of *z/OS MVS Planning: Global Resource Serialization* for information about defining a GRS complex.
2. Using RACGLIST class requires more space for the RACF database. However it ensures that results of any of the following requests are consistent across all systems sharing the database:
   - RACROUTE REQUEST=LIST,GLOBAL=YES
   - Propagated SETROPTS RACLIST command
   - Propagated SETROPTS RACLIST REFRESH command
3. Depending on how your installation's database is set up, it may take less processing time and I/O to read the stored RACLIST results from the RACGLIST profiles than to retrieve the original discrete and generic profiles for the class name. RACGLIST processing may improve startup time and system availability during restarts.
4. If you are using RACGLIST support and your database is being shared by two or more MVS systems, be sure that SYSZRAC2 is *not* in the SYSTEMS exclusion list in SYS1.PARMLIB.

See *z/OS SecureWay Security Server RACF Diagnosis Guide* for a detailed description of RACLIST processing when the RACGLIST class is active.

Administering the Use of Operator Commands

You can control who can issue MVS and JES operator commands regardless of their point of entry. This includes, for example, commands issued at MCS consoles, inline within batch JCL, through SVC 34, or through extended console support.

You can use RACF to authorize the following:

- For MCS consoles, you can authorize individual commands, as well as command groups, to individual operators, groups of operators, or to the consoles.
- For commands issued from NJE nodes and RJE workstations, you can authorize the node or workstation to individual commands or groups of commands.

In addition, the installation can use generic profiles to define groups of commands. If RACF is not used, the system defines the groups of commands. For more information on using MVS and JES to perform command authority checking, see one of the following books:

- For MVS system commands, see *z/OS MVS Planning: Operations*.
- For JES2 commands, see *z/OS JES2 Initialization and Tuning Guide*.
- For JES3 commands, see *z/OS JES3 Initialization and Tuning Guide*.

You can use RACF to perform authority checking for all commands. However, commands issued from locally attached JES3 consoles are checked using JES3's authority, not the operator's authority. In practice, that would probably limit you to just auditing those commands.

"Authorizing the Use of Operator Commands" on page 277 describes how you can use RACF to provide command authority checking. *z/OS JES3 Initialization and Tuning Guide* describes how to use JES to provide command authority checking.

Note: If SDSF is installed on your system, OPERCMDS profiles control which action characters and overtypeable fields users can enter on SDSF panels. For complete information on creating OPERCMDS profiles for use with SDSF, see *z/OS SDSF Operation and Customization*.

Authorizing the Use of Operator Commands

You can control which groups of users (system programmers and operators) can issue commands. You can use RACF to authorize or restrict users from entering some or all commands, or specific variations of commands, or the consoles from which commands can be entered.

To control the use of operator commands, create profiles in the appropriate RACF classes that enable RACF command authorization. The specific RACF classes that you must activate depend on the input source that you want to protect. Table 22 lists the RACF classes that must be activated for each input source.

Table 22. RACF Classes Used to Authorize Operator Commands

Source of Commands:	RACF Security Class to Activate:	Procedure:
Operator commands (except when from remote workstations or NJE nodes)	OPERCMDS	"Controlling the Use of Operator Commands" on page 278
Commands from remote workstations (require additional setup)	OPERCMDS, FACILITY	"Commands from RJE Work Stations" on page 482
Commands from NJE nodes (require additional setup)	OPERCMDS, FACILITY	"Commands from NJE Nodes" on page 482
Commands entered through, or generated by, SDSF	OPERCMDS, CONSOLE	"Administering the Use of Operator Commands" on page 276

Command Authorization in an MCS Sysplex

If a critical system problem occurs in a multiple console support (MCS) sysplex, operators must issue commands to correct the problem. This problem may inhibit access to the RACF database, so MCS saves the security environment in the security object (ENVR) and uses it to perform authorization processing against the OPERCMDS profiles. This way, no access to the RACF database is required at the time.

In order to accomplish this, RACF must be able to successfully process the security object (ENVR) indicated by the ENVRIN value of the RACROUTE REQUEST=VERIFY macro used for OPERCMDS authorization processing. In a sysplex having mixed external security managers, this means that commands routed to systems that use RACF must be issued from systems that use RACF. Therefore, the installation must define MCS consoles so that at least one console attached to a system using RACF is available to issue commands to another system using RACF.

This also means that no refreshing of the list of groups is done. The user ID associated with the MCS console must be reinitialized whenever its user and group data or connections are changed. See *z/OS MVS Planning: Operations* for more details on MCS command authority checking and how to refresh the security environment.

The user and group names are not verified against the database when the security environment is used from another system. All systems in a sysplex should use the

same RACF database. This will provide consistent user, group, and OPERCMD profiles and will ensure accurate authorization checking. In addition, definitions for security categories (members of the CATEGORY profile in the SECDATA general resource class) are likely to cause problems if all systems do not use the same RACF database.

Controlling the Use of Operator Commands

To control the use of operator commands, do the following:

1. Ask your system programmer for the following information:
   a. The subsystem and resource name associated with the command to be authorized. The resource names are described in the following books:
      - For MVS system commands, see *z/OS MVS Planning: Operations*.
      - For JES2 commands, see *z/OS JES2 Initialization and Tuning Guide*.
      - For JES3 commands, see *z/OS JES3 Initialization and Tuning Guide*.
      - For RACF operator commands, you need to define profiles in the OPERCMDS class to control authorization.

Attention

If you don't define profiles for them, these commands cannot be protected. Anyone at a master console or a console with system authority would be able to use these commands. However, except for the DISPLAY command, which does no additional authority checking, these commands check the console's attributes if no profile is found and can still fail the request. For the DISPLAY command, you should specify READ authority.

Table 23. RACF Operator Command Profiles: Naming Conventions

Command	Resource Name
DISPLAY	*subsystem-name*.DISPLAY.SIGNON Any profile in the OPERCMDS class covering this resource name protects the DISPLAY command, for example: `RDEFINE OPERCMDS RACF.DISPLAY.SIGNON`
RESTART	*subsystem-name*.RESTART Any profile in the OPERCMDS class covering this resource name protects the RESTART command, for example: `RDEFINE OPERCMDS RACF.RESTART`
SET	*subsystem-name*.SET.AUTOAPPL *subsystem-name*.SET.AUTODIRECT *subsystem-name*.SET.AUTOPWD *subsystem-name*.SET.INCLUDE *subsystem-name*.SET.JESNODE *subsystem-name*.SET.LIST *subsystem-name*.SET.PWSYNC *subsystem-name*.SET.TRACE To protect the SET LIST command, for example: `RDEFINE OPERCMDS RACF.SET.LIST` **Note:** No OPERCMDS authority check is performed if the SET command is issued from a RACF parameter library member.

Table 23. RACF Operator Command Profiles: Naming Conventions (continued)

Command	Resource Name
SIGNOFF	*subsystem-name*.SIGNOFF Any profile in the OPERCMDS class covering this resource name protects the SIGNOFF command, for example: `RDEFINE OPERCMDS RACF.SIGNOFF`
STOP	*subsystem-name*.STOP Any profile in the OPERCMDS class covering this resource name protects the STOP command, for example: `RDEFINE OPERCMDS RACF.STOP`
TARGET	*subsystem-name*.TARGET.DESCRIPTION *subsystem-name*.TARGET.LIST *subsystem-name*.TARGET.LOCAL *subsystem-name*.TARGET.NODE *subsystem-name*.TARGET.OPERATIVE **Note:** TARGET.OPERATIVE also protects the DELETE and DORMANT operands. *subsystem-name*.TARGET.PREFIX *subsystem-name*.TARGET.PROTOCOL *subsystem-name*.TARGET.PURGE *subsystem-name*.TARGET.WDSQUAL *subsystem-name*.TARGET.WORKSPACE To protect the TARGET LIST command, for example: `RDEFINE OPERCMDS RACF.TARGET.LIST` **Note:** No OPERCMDS authority check is performed if the TARGET command is issued from a RACF parameter library member.

Table 24. RACF TSO Commands Entered as Operator Commands: Naming Conventions

Command	Resource Name
ADDGROUP or AG	*subsystem-name*.ADDGROUP
ADDSD or AD	*subsystem-name*.ADDSD
ADDUSER or AU	*subsystem-name*.ADDUSER
ALTDSD or ALD	*subsystem-name*.ALTDSD
ALTGROUP or AG	*subsystem-name*.ALTGROUP
ALTUSER or ALU	*subsystem-name*.ALTUSER
CONNECT or CO	*subsystem-name*.CONNECT
DELDSD or DD	*subsystem-name*.DELDSD
DELGROUP or DG	*subsystem-name*.DELGROUP
DELUSER or DU	*subsystem-name*.DELUSER
LISTDSD or LD	*subsystem-name*.LISTDSD
LISTGRP or LG	*subsystem-name*.LISTGRP
LISTUSER or LU	*subsystem-name*.LISTUSER
PASSWORD or PW	*subsystem-name*.PASSWORD
PERMIT or PE	*subsystem-name*.PERMIT
RACLINK	*subsystem-name*.RACLINK

Table 24. RACF TSO Commands Entered as Operator Commands: Naming Conventions (continued)

Command	Resource Name
RALTER or RALT	*subsystem-name*.RALTER
RDEFINE or RDEF	*subsystem-name*.RDEFINE
RDELETE or RDEL	*subsystem-name*.RDELETE
REMOVE or RE	*subsystem-name*.REMOVE
RLIST or RL	*subsystem-name*.RLIST
SEARCH or SR	*subsystem-name*.SEARCH
SETROPTS or SETR	*subsystem-name*.SETROPTS

Notes:

1) RACF first checks that the operator issuing the TSO command is defined to RACF and if not, an error message is issued. If the operator is defined to RACF, a check is made to a profile in the OPERCMDS class to determine if the user ID has authority to issue the TSO command as an operator command. If the OPERCMDS class is not active, or if no OPERCMDS profile exists, the user will be allowed to issue the command as an operator command.
2) Existing command authorization is still enforced. For example, you must be a SPECIAL user to issue the SETROPTS INITSTATS command.
3) READ access is required to all the resource names shown in Table 24 on page 279 with the exception of SETROPTS. If SETROPTS LIST is issued with no other operands, READ access is sufficient. However, if any other SETROPTS option is issued, with or without also specifying LIST, UPDATE access is required.
4) If your installation renames any RACF TSO commands, they are still protected under the resource names shown in Table 24 on page 279. For example, if you renamed ADDGROUP as ADDBUNCH, RACF would still use *subsystem-name.*ADDGROUP as the resource name.

Other examples of defining profiles in the OPERCMDS class follow:

```
RDEFINE OPERCMDS RACF.SIGNOFF.** UACC(NONE)
PERMIT  RACF.SIGNOFF.** CLASS(OPERCMDS) ID(DJONES) ACCESS(UPDATE)

RDEFINE OPERCMDS RACF.DISPLAY.SIGNON.** UACC(NONE)
PERMIT  RACF.DISPLAY.SIGNON.** CLASS(OPERCMDS) ID(DJONES) ACCESS(UPDATE)
```

The base command or resource names are SIGNOFF and DISPLAY.SIGNON. Although SIGNON is not required at the console (because it is the default), it must be specified in the resource name to protect the DISPLAY command.

The RVARY command cannot be protected by profiles in the OPERCMDS class. This is intentional during recovery; RACF must not be allowed to attempt to access the database. The RVARY command is always protected by an operator prompt, regardless of whether it is entered from TSO or as an operator command.

b. The UACC to be associated with the command.
c. The user IDs of the operators or the group names of the groups of operators to whom you want to grant authority.
d. For each operator or group of operators:

- The access authority to be assigned to the operator or group of operators.
- Any restrictions on which consoles the operators must be using when issuing certain commands. To do this, create profiles for consoles in the CONSOLE class and then specify the WHEN(CONSOLE) operand on the PERMIT command. See "Conditional Access Lists for General Resource Profiles" on page 206.

Note: To authorize a console to a command or group of commands, create a RACF user profile for the console and place the console's user ID (or a RACF group to which the user ID is connected) in the access list of the OPERCMDS profile.

2. Use the RDEFINE command to create profiles for the commands:

   ```
   RDEFINE OPERCMDS profile-name UACC(NONE)
   ```

   where *profile-name* is:

   ```
   subsystem-name.command[.qualifier]
   ```

   where:

subsystem-name	is the name of the processing environment of the command (such as MVS, JES2, JES3, or RACF)
command	is the name of the command
qualifier	is the type of object the command specifies (JOB or SYS, for example) or an operand of the command (LIST, for example)

   In these examples, you would first issue SETROPTS GENERIC(OPERCMDS) to turn generics on for the OPERCMDS class and then issue SETROPTS REFRESH:

   ```
   RDEFINE OPERCMDS JES3.CALL.**     UACC(NONE)
   RDEFINE OPERCMDS RACF.TARGET.LIST UACC(NONE)
   ```

   Notes:

   a. When an operator issues a command that the subsystem doesn't recognize, the subsystem checks for a profile named *subsystem-name*.UNKNOWN. To handle these commands, create a profile named:
      - MVS.UNKNOWN with UACC(READ) for MVS
      - JES2.UNKNOWN or JES3.UNKNOWN with UACC(NONE) for JES
      - RACF.UNKNOWN with UACC(NONE) for RACF

      Your security policy may require auditing of all commands issued, even if they are not valid on your system. You can audit these commands by specifying AUDIT(ALL) on these profiles.

3. For each controlled command, grant access to the users or groups who need to use it:

   ```
   PERMIT profile-name CLASS(OPERCMDS) ID(user or group ACCESS(access-authority)
   ```

   For example:

   ```
   PERMIT JES3.CALL.DSP.** CLASS(OPERCMDS) ID(OPER7 OPER24) ACCESS(UPDATE)
   ```

4. When you are ready to start controlling access to commands based on the profiles you have defined, activate the OPERCMDS class:

```
SETROPTS CLASSACT(OPERCMDS) RACLIST(OPERCMDS)
```

Example of Controlling Who Can Issue MVS Commands

The following example shows how to use the OPERCMDS class to control who can display and cancel jobs in your installation.

Suppose you want to let anyone display jobs but you want to restrict the task of cancelling jobs to a group of MVS operators. All of the MVS operators you want to authorize have RACF-defined user IDs connected to a group called OPERGRP.

According to *z/OS MVS Planning: Operations*, to authorize a user to issue the MVS DISPLAY JOBS command, you must give the user READ access to a resource named MVS.DISPLAY.JOB in the OPERCMDS class. To authorize a user to issue the MVS CANCEL command for all jobs, you must give the user UPDATE access to a resource named MVS.CANCEL.JOB.** in the OPERCMDS class.

To grant these authorizations, enter:

```
SETROPTS GENERIC(OPERCMDS) REFRESH

RDEFINE  OPERCMDS MVS.DISPLAY.JOB UACC(READ)
RDEFINE  OPERCMDS MVS.CANCEL.JOB.** UACC(NONE)

PERMIT   MVS.CANCEL.JOB.** CLASS(OPERCMDS) ID(OPERGRP) ACCESS(UPDATE)

SETROPTS CLASSACT(OPERCMDS)
SETROPTS RACLIST(OPERCMDS) REFRESH
SETROPTS GENERIC(OPERCMDS) REFRESH
```

Now, anyone can display a job, but only the operators in OPERGRP can cancel a job.

Controlling the Use of Remote Sharing Functions

You can control the use of most RACF remote sharing facility (RRSF) functions through profiles in the RRSFDATA class.

When you set up RRSF, you need to think on a network level, rather than on a node level. For example, suppose users can't define associations or synchronize passwords on NODE1, but they can on NODE2. Susan has user IDs on NODE1 and NODE2. On NODE2, she can create an association with her NODE1 user ID and synchronize password changes she makes on NODE2 with her NODE1 user ID. She gets an association on NODE1 even though NODE1 doesn't allow her to set them up, and gets 1-way password synchronization to her NODE1 user ID, even though NODE1 doesn't allow password synchronization.

Controlling Access to the RACLINK Command

You can control whether users can issue the RACLINK command to establish user ID associations and enable password synchronization.

Controlling the Use of the RACLINK DEFINE Operand

RACLINK commands that specify the DEFINE operand are subject to a security check to determine if the command issuer is authorized to issue RACLINK DEFINE to the specified node. Because use of the DEFINE operand is controlled at a node level, the local node and all target nodes defined to it must have appropriate profiles.

To allow a user ID association to be made to a specific node, create a profile in the RRSFDATA class to protect a resource called RACLINK.DEFINE.*node*, where *node* is the node name.

The request issuer needs to have READ access to the resource. The request will fail if:

1. The RRSFDATA class is inactive.
2. There is no RRSFDATA profile protecting RACLINK.DEFINE.*node* for the specified node.
3. The command issuer is not properly authorized to the resource.

Controlling the Use of the RACLINK PWSYNC Operand

RACLINK commands that specify the PWSYNC operand are subject to a security check to determine if the command issuer is authorized to request password synchronization with user IDs that are on the specified node. Because password synchronization is controlled at a node level, the local node and all target nodes defined to it must have appropriate profiles.

To allow password synchronization to occur with user IDs on a specific node, create a profile in the RRSFDATA class to protect a resource called RACLINK.PWSYNC.*node*, where *node* is the node name.

When you issue a request to create an association with password synchronization, you need to have:

- The authority to issue the RACLINK DEFINE command with PWSYNC specified
- READ access to the RACLINK.DEFINE.*node* and RACLINK.PWSYNC.*node* resources

The request will fail if:

1. The RRSFDATA class is inactive.
2. There is no RRSFDATA profile protecting RACLINK.PWSYNC.*node* for the specified node.
3. You do not have READ access to the RACLINK.DEFINE.*node* and RACLINK.PWSYNC.*node* resources.

Controlling Password Synchronization

To activate password synchronization, issue the SETPWSYNC command. See *z/OS SecureWay Security Server RACF Command Language Reference* for more information. The PWSYNC profile in the RRSFDATA class further controls password synchronization. With this profile, you can enable or disable password synchronization for users with PEER PWSYNC associations on an RRSF node.

A user must be permitted with READ access so that PWSYNC requests for the user are processed successfully. Or, if no permit has been done for the user (or a group the user connected to), you can give the PWSYNC profile a UACC of READ, which enables password synchronization for all users who have approved PEER associations with PWSYNC. If the RACF RRSFDATA class is not active, or the PWSYNC profile has not been defined, password synchronization will not occur even for the users with established associations.

The PWSYNC profile does not initiate password synchronization. Users must have approved associations with password synchronization enabled. Using this profile, you can decide which users password changes should be synchronized.

General Resources

To enable password synchronization for users with RACLINK PEER PWSYNC associations and disable automatic password direction:

```
SET PWSYNC NOAUTOPWD
```

You can also use the PWSYNC profile to control password synchronization at a system level. For example, to turn off password synchronization without having to delete all of the existing user ID associations, delete the PWSYNC profile, or, give it a UACC of NONE with no users on the access list. To turn off password synchronization, you can specify SET NOPWSYNC. See *z/OS SecureWay Security Server RACF Command Language Reference* for details.

Controlling the Use of the AT Operand

Commands that specify the AT operand are subject to a security check to determine if the command issuer is authorized to direct RACF TSO commands to the specified node. Because command direction is controlled at a node level, the local node and all target nodes defined to it must have appropriate profiles.

To allow command direction to a specific node, create a profile in the RRSFDATA class to protect a resource called DIRECT.*node*, where *node* is the node name.

The request issuer needs to have READ access to the resource. The request will fail if:

1. The RRSFDATA class is inactive.
2. There is no RRSFDATA profile protecting DIRECT.*node* for the specified node.
3. The command issuer is not properly authorized to the resource.

For information on implementing command direction using the AT operand,see “Directing Commands Using the AT Option” on page 364.

Controlling the Use of the ONLYAT Operand

Commands that specify the ONLYAT operand are subject to a security check to determine if the command issuer is authorized as follows:

- The command issuer and the target user ID must be SPECIAL.
- No user ID association is required if the target user ID is the same as the command issuer. The user IDs can be on different nodes.
- If the target user ID is different from the command issuer, a user ID association between the command issuer and the target user ID is required. This prevents a SPECIAL user from unauthorized use of another remote SPECIAL user ID.

For information on implementing command direction using the AT operand, see “Directing Commands Using the ONLYAT Option” on page 367.

Controlling Automatic Direction

You can control automatic direction of commands, passwords, and application updates through profiles in the RRSFDATA class. These are also controlled at a system level through the AUTODIRECT, AUTOPWD, and AUTOAPPL operands of the SET command. See *z/OS SecureWay Security Server RACF Command Language Reference* for details.

Controlling Automatic Direction of Commands

Profiles in the RRSFDATA class control which commands are automatically directed to which nodes. The resource name format is:

```
AUTODIRECT.target-node.classname.command-name
```

where:

target-node Is the remote node where the command is to be directed.

classname Is the class name associated with the command issued. The class name can be USER, GROUP, DATASET, or any general resource class defined in the class descriptor table (CDT).

command-name
Is the name of the command issued.

The use of these profiles provides security for automatic command direction. An authorization check is made against these resource names to determine if the user is allowed to automatically direct the specified command. The command is directed to the remote node if:

- The RRSFDATA class has been activated.
- SET AUTODIRECT is in effect.
- There is a profile for the resource name associated with the command.
- The command issuer has at least READ access to that resource.

Table 25 lists the resource name for each RACF command that can be used with automatic command direction.

Table 25. Automatic Command Direction: Resource Names

Command	Class	Resource Name
ADDUSER or AU	USER	AUTODIRECT.*target-node*.USER.ADDUSER
ALTUSER or ALU	USER	AUTODIRECT.*target-node*.USER.ALTUSER
CONNECT or CO	USER	AUTODIRECT.*target-node*.USER.CONNECT
DELUSER or DU	USER	AUTODIRECT.*target-node*.USER.DELUSER
PASSWORD or PW	USER	AUTODIRECT.*target-node*.USER.PASSWORD
REMOVE or RE	USER	AUTODIRECT.*target-node*.USER.REMOVE
ADDGROUP or AG	GROUP	AUTODIRECT.*target-node*.GROUP.ADDGROUP
ALTGROUP or ALG	GROUP	AUTODIRECT.*target-node*.GROUP.ALTGROUP
DELGROUP or DG	GROUP	AUTODIRECT.*target-node*.GROUP.DELGROUP
ADDSD or AD	DATASET	AUTODIRECT.*target-node*.DATASET.ADDSD
ALTDSD or ALD	DATASET	AUTODIRECT.*target-node*.DATASET.ALTDSD
DELDSD or DD	DATASET	AUTODIRECT.*target-node*.DATASET.DELDSD
PERMIT or PE	any general resource class or DATASET	AUTODIRECT.*target-node.classname*.PERMIT
RALTER or RALT	any general resource class	AUTODIRECT.*target-node.classname*.RALTER
RDEFINE or RDEF	any general resource class	AUTODIRECT.*target-node.classname*.RDEFINE
RDELETE or RDEL	any general resource class	AUTODIRECT.*target-node*.classname.RDELETE
SETROPTS or SETR	none (use RACF)	AUTODIRECT.*target-node*.RACF.SETROPTS

Notes:

1. To activate automatic command direction, issue the SET AUTODIRECT command. See “Automatic Direction” on page 367 and *z/OS SecureWay Security Server RACF Command Language Reference* for more information.
2. Automatic command direction occurs only at the command level. You cannot direct a command operand or segment information for a command. For example, if you direct the ADDUSER command, you direct all ADDUSER commands, including the TSO, DFP, and OPERPARM segment information. You cannot specify automatic command direction for only the TSO segment information in the ADDUSER command.
3. You can use generic profiles to define these profiles. No commands will be directed if the RRSFDATA class is inactive or if no RRSFDATA profiles that protect AUTODIRECT exist.
4. These profiles are only checked on the node where the command was issued. Once the command is directed to another node, no authorization check is made against these profiles on the receiving node.
5. Profiles for turning on automatic direction of passwords and application updates are similar. Therefore, using * for the command names will turn on these functions, too.
6. If your installation renames any RACF TSO commands, they are still protected under the resource names shown in Table 25 on page 285. For example, if you renamed ADDGROUP as ADDBUNCH, RACF would still use AUTODIRECT.*target-node*.GROUP.ADDGROUP as the resource name.

Sample Automatic Command Direction Profiles: You can activate automatic direction of commands without activating automatic direction of application updates by using SET AUTODIRECT NOAUTOAPPL. You can also turn off password propagation by issuing the SET AUTODIRECT NOAUTOPWD command. See *z/OS SecureWay Security Server RACF Command Language Reference* for details.

Some examples of using profiles to control automatic command direction follow. For each example, assume that no other profiles beginning with AUTODIRECT are present in the RRSFDATA class.

- To disable automatic command direction for TAPEVOL profiles and direct all other commands to all remote nodes:

```
AUTODIRECT.*.TAPEVOL.*     UACC(NONE), no users on access list
AUTODIRECT.**              UACC(READ), no users on access list
```

- To direct ADDUSER commands issued by BOB to all remote nodes:

```
AUTODIRECT.*.USER.ADDUSER  UACC(NONE), BOB on access list with READ access
```

- To disable automatic command direction for TAPEVOL and RRSFDATA profiles and direct all other commands to all remote nodes:

```
AUTODIRECT.*.TAPEVOL.*     UACC(NONE), no users on access list
AUTODIRECT.*.RRSFDATA.*    UACC(NONE), no users on access list
AUTODIRECT.**              UACC(READ), no users on access list
```

- To enable automatic command direction only to NODE1 for the USER and GROUP classes:

```
AUTODIRECT.NODE1.USER.*    UACC(READ), no users on access list
AUTODIRECT.NODE1.GROUP.*   UACC(READ), no users on access list
AUTODIRECT.**              UACC(NONE), no users on access list
```

Controlling Automatic Direction of Passwords

Profiles in the RRSFDATA class control which passwords get automatically directed to which nodes. The format for the resource names is:

AUTODIRECT.*target-node*.USER.PWSYNC

where:

target-node Is the remote node where the command is to be directed

These profiles provide security for automatic password direction. An authorization check is made against these resource names to determine if the user's password can be synchronized automatically. The password change will be directed to the remote node if:

- SET AUTOPWD is in effect.
- The RRSFDATA class has been activated.
- There is a profile to cover the resource name:

  AUTODIRECT.*target-node*.USER.PWSYNC
- The user changing the password has at least READ access to that resource.

You can use generic profiles to define these profiles. If the RRSFDATA class is inactive or if no RRSFDATA profile that protects AUTODIRECT.*target-node*.USER.PWSYNC exists, password changes are not directed automatically.

The RRSFDATA profile that protects AUTODIRECT.*target-node*.USER.PWSYNC is only checked on the node where the password is originally changed. Once the password change is directed to another node, no authorization check is made on the receiving node.

Sample Automatic Password Direction Profiles: Some examples of using profiles to control automatic password direction follow. For each example, assume that no other profiles beginning with AUTODIRECT are present in the RRSFDATA class.

- To enable password synchronization for users with RACLINK PEER PWSYNC associations:

  ```
  PWSYNC                      UACC(READ)
  AUTODIRECT.*.USER.PWSYNC    UACC(NONE)
  ```

  AUTODIRECT.*.USER.PWSYNC is not required, but if you have other profiles that protect AUTODIRECT, this prevents automatic password direction.
- To enable automatic password direction for users without RACLINK associations:

  ```
  PWSYNC                      UACC(NONE)
  AUTODIRECT.*.USER.PWSYNC    UACC(READ)
  ```
- To enable automatic password direction for users without RACLINK associations to node MVS1:

  ```
  PWSYNC                      UACC(NONE)
  AUTODIRECT.MVS1.USER.PWSYNC UACC(READ)
  ```

Controlling Automatic Direction of Application Updates

Profiles, including generic profiles, in the RRSFDATA class control which application updates get automatically directed to which nodes. The format for the resource names for USER, GROUP, class descriptor table (CDT) classes, and some DATASET updates is:

AUTODIRECT.*target-node.classname*.APPL

where:

target-node Is the remote node the update is to be propagated to

classname Is the class name associated with the update. This is USER, GROUP, any general resource class, or DATASET for updates not covered by the AUTODASD and AUTOTAPE profiles.

The formats when you are using this syntax for automatic direction of application updates in the DATASET class are:

AUTODIRECT.*target-node*.DATASET.APPL

AUTODASD.*target-node*.DATASET.APPL

AUTOTAPE.*target-node*.DATASET.APPL

where:

target-node Is the remote node the update is to be propagated to

Use AUTODIRECT.*target-node*.DATASET.APPL to control automatic direction of application updates for DATASET when the request is RACROUTE REQUEST=EXTRACT, RACXTRT, or ICHEINTY.

Use AUTODASD when:

- The request is a RACROUTE REQUEST=DEFINE or a RACDEF.
- The CLASS value is set to, or defaults to, DATASET.
- The DSTYPE value is not T.

Use AUTOTAPE when:

- The request is a RACROUTE REQUEST=DEFINE or a RACDEF.
- The CLASS value is set to, or defaults to, DATASET.
- The DSTYPE value is T.

These profiles provide security for automatic direction of application updates. An authorization check is made against these resource names to determine if the user is allowed to make these updates. The application updates are directed to the remote node if:

- Automatic direction has been activated using SET AUTOAPPL.
- The RRSFDATA class is active.
- There is a profile to cover the resource name AUTODIRECT.*target-node.classname*.APPL, AUTODASD.*target-node*.DATASET.APPL, or AUTOTAPE.*target-node*.DATASET.APPL.
- The user directing the application update has at least READ access to that resource.

The RRSFDATA profile that protects AUTODIRECT.*target-node.classname*.APPL, AUTODASD.*target-node*.DATASET.APPL, or AUTOTAPE.*target-node*.DATASET.APPL is only checked on the node where the update originates. Once the update is propagated to another node, no AUTODIRECT authorization check is made on the receiving node.

Suppression of Private Key Information Propagation: When automatic direction of application updates is enabled, RACF database changes initiated by RACDCERT are propagated to other systems. These changes include additions, alterations, and deletions of certificates. However, private key information contained in the following fields of general resource profiles in DIGTCERT class is not propagated:

CERTPRVK Private key

CERTPRVS Private key size

CERTPRVT Private key type

RACF does not store the private key associated with a certificate. Instead, RACF stores the Integrated Cryptographic Service Facility (ICSF) key token. The private key itself is stored in the ICSF public key data set (PKDS) and is encrypted under the master key of the system. Since the automatic direction of application updates propagates information to remote systems that are likely to have different public key data sets and different master keys, private key information from the originating node is unusable.

Sample Automatic Direction of Application Updates: Some examples of using profiles to control automatic application update direction follow. For each example, assume that no other profiles beginning with AUTODIRECT are present in the RRSFDATA class.

- To disallow both automatic direction of commands and automatic direction of application updates for TAPEVOL and RRSFDATA profiles, disallow automatic updates for TAPEVOL and RRSFDATA profiles, disallow automatic direction of application updates for all DATASET profiles, and allow all other updates to propagate to all remote nodes:

```
AUTODIRECT.*.TAPEVOL.*     with UACC(NONE) and no users on access list
AUTOTAPE.*.DATASET.*       with UACC(NONE) and no users on access list

AUTODIRECT.**              with UACC(READ) and no users on access list
AUTODASD*.**               with UACC(READ) and no users on access list
```

- The following RRSFDATA profiles allow both automatic direction of commands and automatic direction of application updates only to NODEA for the USER and GROUP classes. In this example, no AUTOTAPE or AUTODASD profiles are defined. This gives the same results as defining the profiles with a UACC of NONE (updates are not propagated):

```
AUTODIRECT.NODEA.USER.*    with UACC(READ) and no users on access list
AUTODIRECT.NODEA.GROUP.*   with UACC(READ) and no users on access list
```

Establishing Security for the RACF Parameter Library

The RACF parameter library should be protected appropriately through the use of a DATASET profile. No OPERCMDS authority check is performed for commands issued from within a RACF parameter library member. Commands issued from within a RACF parameter library run with the authority of the RACF subsystem address space.

Controlling Message Traffic

You can control whether users can receive messages sent with the TSO SEND command.

Notes:

1. When the SMESSAGE class is active and a profile does not exist for the specified user, the message request completes normally.
2. When the SECLABEL class is active, the receiver of the message must pass the security label authorization check based on the receiver's current security label and the security label of the message (which was set by the sender's current security label at the time that the sender sent the message.)

To control message traffic, do the following:

1. For each user for which you wish to control message traffic, create a profile in the SMESSAGE class:

```
RDEFINE SMESSAGE receiving-userid UACC(NONE)
```

2. Give users the appropriate access authority:

   ```
   PERMIT receiving-userid CLASS(SMESSAGE) ID(sending-userid-or-group)
          ACCESS(access-authority)
   ```

   where *access-authority* is one of the following:

   NONE Prevents users and groups in the access list from sending messages to the user whose ID is the profile name

   READ Allows users and groups in the access list to send messages to the user whose ID is the profile name.

3. When you are ready to start using the security provided by these profiles, activate both the SMESSAGE class and SETROPTS RACLIST processing for the class. SETROPTS RACLIST processing helps ensure high performance when access authorities are checked. You can do these actions in one command:

   ```
   SETROPTS CLASSACT(SMESSAGE) RACLIST(SMESSAGE)
   ```

   Notes:

   a. Any time you make a change to an SMESSAGE profile, you must also refresh SETROPTS RACLIST processing for the SMESSAGE class for the change to take effect. For example:

      ```
      SETROPTS RACLIST(SMESSAGE) REFRESH
      ```

   b. This facility is intended primarily as an audit mechanism for installations in a B1 level of security. By itself, the SMESSAGE class does not control all means of communication among users on the system.

4. Security label checking for messages is available through the DIRAUTH class. This is primarily a B1 security function. See *MVS/ESA SP V5 Planning: B1 Security (GC28–1440)* for more information.

Controlling the Opening of VTAM ACBs

With VTAM 3.3 and beyond, you can control which users can open the application control block (ACB) indicated by a VTAM application program when the user is not running an APF-authorized program or command processor. For more information, see *z/OS Communications Server: SNA Programming*.

To do this, take the following steps:

1. Ask your VTAM system programmer for the following information:
   - The names of the VTAM application programs whose use is to be controlled
   - The names of RACF-defined users and groups who are to have access to those programs.
2. Create profiles in the VTAMAPPL class:

   ```
   RDEFINE VTAMAPPL acb-name UACC(NONE)
   ```

   where *acb-name* is the ACBNAME value on the APPL statement that applies to this ACB. (An ACB name is also called an LU name or a VTAM application name.)

3. Give users and groups the appropriate access authority:

   ```
   PERMIT acb-name CLASS(VTAMAPPL)
          ID(userid or groupname)
          ACCESS(access-authority)
   ```

   where *access-authority* is one of the following:

NONE	Prevents users from opening the ACB
READ	Allows users to open the ACB
UPDATE	Is the same as READ
CONTROL	Is the same as READ
ALTER	Allows READ access, and also allows users to change the profile (if it is a discrete profile).

4. When you are ready to start using the protection defined in the profiles, activate both the VTAMAPPL class and SETROPTS RACLIST processing for the class. You can do these two actions in one command:

   ```
   SETROPTS CLASSACT(VTAMAPPL) RACLIST(VTAMAPPL)
   ```

   Note: Any time you make a change to a VTAMAPPL profile, you must also refresh SETROPTS RACLIST processing for the VTAMAPPL class for the change to take effect.

RACF and PSF (Print Services Facility)

If Print Services Facility for OS/390 is installed, and the SECLABEL class is active, you can control how printed output is affected as follows:

- The separator pages (the job header and trailer) are always printed with the PSF identification label associated with the security label of the user data and cannot be falsified by a user. Printing of separator pages can be suppressed by the operator or the system programmer, but not by the print-job submitter.
- Data page labeling is in effect for the user data pages. This means that the PSF identification label associated with the security label of the user data is printed on all pages of printed output. By granting READ access to the PSF.DPAGELBL resource in the PSFMPL class, you can selectively allow users to stop the printing of PSF identification labels in printed output.
- On certain printers, end user data can be printed only in a system-defined user printable area. This function allows the PSF identification label to print end use data outside the system-defined user printable area.

Note: PSF print labeling support depends on the type of printer being used.

For specific information on PSF printers and information on using RACF to provide security for PSF, see *PSF: Security Guide*.

Auditing When Users Receive Message Traffic

You can audit when users receive data sent with the TSO SEND command or the TPUT macro, or received using the LISTBC command.

The following procedure sets up this auditing:

1. A user with the SPECIAL attribute activates the DIRAUTH class:

   ```
   SETROPTS CLASSACT(DIRAUTH)
   ```

   Note: No profiles are needed in the DIRAUTH class.
2. A user with the AUDITOR attribute requests that auditing be done each time a user receives data:

   ```
   SETROPTS LOGOPTIONS(ALWAYS(DIRAUTH))
   ```

To stop auditing the DIRAUTH class, enter:

```
SETROPTS LOGOPTIONS(NEVER(DIRAUTH))
```

RACF and APPC

RACF provides support to APPC in several ways:

- User verification during APPC transactions
- Support of persistent verification “signed_on_from” lists
- Protection of APPC transaction programs
- Protection of APPC server IDs (APPCSERV)
- LU security capabilities

For more information about APPC, see *z/OS MVS Planning: APPC/MVS Management*.

User Verification during APPC Transactions

When APPC/MVS receives a request to allocate an APPC transaction program, it uses RACF to verify the user ID and password (if any) to be associated with that transaction.

The verification also includes checking the user’s authority to use both the partner LU and the local LU to which the transaction request was routed.

Partner LU as Port of Entry (POE)

You can use the APPCPORT general resource class to protect the port of entry (POE). There are two possible formats for the resource name in the APPCPORT class:

- If the APPC LU definition has enabled network-qualified names support (by specifying the NQN option on the LUADD statement), the format for the resource name is:

  netid.luname

  where:

netid	Is a network name consisting of 1–8 characters. The first character must be alphabetic.
luname	Is an LU name consisting of 1–8 characters.

- If APPC network-qualified names support is not enabled, the format for the resource name is:

  luname

  where:

luname	Is an LU name consisting of 1–8 characters. The first character must be alphabetic.

Failure to specify the correct LU name format could result in a security exposure. For more detailed information, see *z/OS MVS Planning: APPC/MVS Management*.

Local LU Name as Application (APPL)

RACF uses the APPL class to control the attach request. For more detailed information, see *z/OS MVS Planning: APPC/MVS Management*.

Protection of APPC/MVS Transaction Programs (TPs)

The security administrator can define profiles to the APPCTP class to protect APPC applications in which the outbound transaction program issues an allocate request

for an inbound transaction program on MVS. For more detailed information, see *z/OS MVS Planning: APPC/MVS Management*.

Example:

The following example illustrates how you can use the RDEFINE command to define a transaction program profile named FINANC1.SMITH.ACCTPAYABLETP and specify a UACC of READ. A UACC of READ allows all users to access the transaction program and their keys.

```
RDEFINE APPCTP FINANC1.SMITH.ACCTPAYABLETP UACC(READ)
```

You can protect a transaction program by specifying a UACC of NONE. You can then create an access list that contains only those users who need access. The following example shows how you can define a transaction program profile named FINANC1.SMITH.ACCTPAYABLETP and give it a UACC of NONE:

```
RDEFINE APPCTP FINANC1.SMITH.ACCTPAYABLETP UACC(NONE)
```

After you protect the transaction program with a UACC of NONE, you can use the PERMIT command to define entries in the transaction program profile's access list. The following example shows how to use the PERMIT command to create entries in the access list of transaction program profile FINANC1.SMITH.ACCTPAYABLETP for users USERA and USERB, giving them each an access authority of READ:

```
PERMIT FINANC1.SMITH.ACCTPAYABLETP CLASS(APPCTP) ID(USERA USERB) ACCESS(READ)
```

The following example illustrates how you can use the RDEFINE command to define a CPI-C side information profile named TOOLS1.SYS1.SDLU1234 and specify a UACC of READ, which allows all users to read CPI-C side information.

```
RDEFINE APPCSI TOOLS1.SYS1.SDLU1234 UACC(READ)
```

You can protect CPI-C side information by specifying a UACC of NONE. You can then create an access list containing only users who need access. The following example shows how you can define a CPI-C side information profile named TOOLS1.SYS1.SDLU1234 and give it a UACC of NONE:

```
RDEFINE APPCSI TOOLS1.SYS1.SDLU1234 UACC(NONE)
```

After you protect CPI-C side information with a UACC of NONE, you can use the PERMIT command to define entries in the CPI-C side information profile's access list. The following example shows how to use the PERMIT command to create entries in the access list of CPI-C side information profile TOOLS1.SYS1.SDLU1234 for users USERA and USERB, giving them each an access authority of READ:

```
PERMIT TOOLS1.SYS1.SDLU1234 CLASS(APPCSI) ID(USERA USERB) ACCESS(READ)
```

LU Security Capabilities

You can specify the conversation security you want to receive in the APPCLU profile that covers a session.

Conversation Security Options

The conversation security options are stored in the CONVSEC field in the SESSION segment of the APPCLU general resource profile. CONVSEC specifies the information that is sent to the partner LU during BIND, and indicates what conversation security options are acceptable to this LU.

Origin LU Authorization

You can use the APPL general resource class to protect conversations between partner LUs. This support provides the ability to grant or deny access on the basis of the identity of both the user and the LU from which the user's request originated.

An example of how a security administrator would define origin LU authorization is as follows:

```
RDEFINE APPL local-luname UACC(NONE)
```

This command creates a RACF profile for the given LU. The specified UACC in this case would allow no user access to the LU named by *local-luname* without explicitly granted higher access authority.

Next, the security administrator could grant conditional access to a specific RACF-defined user or group whose request originates at a given partner LU with the following:

```
PERMIT local-luname CLASS(APPL) ID(userid)
       ACCESS(READ) ...
       WHEN(APPCPORT(partner-luname))
```

Note: There are two possible formats for the resource name in the APPCPORT class. See "Partner LU as Port of Entry (POE)" on page 292 for additional information.

In this example, you could specify ID(*) to make LU *local-luname* accessible to anyone who is valid on the local system and whose request originates from LU *partner-luname*. Also, this example presupposes that the relevant classes have already been explicitly activated.

Using the WHEN() option puts an entry on the conditional access list of the RACF profile for *local-luname*, allowing *userid* READ access to this LU. This allows *userid* to use the local LUs services, but only when *partner-luname* is the port of entry from which the request originated.

Protection of APPC Server IDs (APPCSERV)

You can use the APPCSERV general resource class to allow authorization checking for a program running in an MVS address space that has identified itself as a server for a specific APPC/MVS transaction program. By protecting a resource in the APPCSERV class for each TP and giving READ access to authorized server IDs, you can ensure that only authorized server programs are allowed to serve a particular TP.

RACF and CICS

If CICS is installed on your system, you can use RACF to provide security for CICS resources. For more information, see *CICS RACF Security Guide*.

RACF and DB2

If DB2 is installed on your system, you can use the DSNR general resource class for controlling access to DB2 subsystems. RACF can control which users can use DB2. If you have multiple DB2 subsystems running, RACF controls which users can use a specific DB2 subsystem. DB2 uses the user's RACF user ID in making its security decisions. Depending on the version of DB2 installed, DB2 can use a user's RACF group connections as secondary authorization IDs for DB2 security

decisions. For information about using the DSNR class, see the appropriate DB2 publication for your installation—for example:

- *DB2 for MVS/ESA Version 4 Administration Guide,* SC26-3265
- *DB2 for OS/390 Version 5 Administration Guide,* SC26-8957
- *DB2 Universal Database for OS/390 Version 6 Administration Guide,* SC26-9003

If you are using DB2 Version 5, or higher, and the *RACF/DB2 external security module* is installed on your system, you can control access to DB2 resources using RACF profiles. See “Chapter 11. Controlling Access to DB2 Objects” on page 387 for more information.

RACF and ICSF

If Integrated Cryptographic Service Facility (ICSF) is installed on your system, you can use RACF to control who can use ICSF cryptographic keys and services.

This control is implemented using the CSFSERV, CSFKEYS, and GCSFKEYS classes. When these classes are used, they must be activated and brought into storage through SETROPTS RACLIST.

RACF and z/OS UNIX

You can use RACF to provide additional security functions and administration for your z/OS UNIX environment. See “Chapter 18. RACF and z/OS UNIX” on page 503.

For complete information on setting up and using RACF in the z/OS UNIX environment, see *z/OS UNIX System Services Planning*.

RACF Support for NDS and Lotus Notes for z/OS

You can map Lotus Notes for z/OS short names and Novell Directory Services for OS/390 (NDS) user names to RACF user IDs. NDS and Lotus Notes for z/OS can determine the RACF user ID for a user who has been authenticated with an application user identity or a digital certificate. Once the application determines a user’s RACF user ID, it may choose to use this identity for authorization checking when accessing traditional system resources, such as data sets. This allows your installation to maintain existing functions, resources, and security while consolidating application servers.

Administering Application User Identities

You manage application user identities, such as Lotus Notes for z/OS short names and Novell Directory Services for OS/390 (NDS) user names, by administering user profiles. Through the ADDUSER, ALTUSER, and DELUSER commands, you can associate a RACF user ID with an application user identity, and you can change and remove that association.

The following table shows the name of the identity segment of the user profile and the name of the user identity field in the identity segment that is used to associate application user identities to RACF user IDs.

Application	Identity segment in the user profile	User identity field
Lotus Notes for z/OS	LNOTES	SNAME

Application	Identity segment in the user profile	User identity field
Novell Directory Services for OS/390	NDS	UNAME

Each RACF user ID can map to both a Lotus short name and an NDS user name, but no user ID can map to more than one Lotus short name or more than one NDS user name. In addition, each Lotus short name can map to only one user ID. Similarly, each NDS user name can map to only one user ID.

The application user identities that you specify in the identity segments of the user profiles must match the user identities defined by the administrators of each application. For example, the SNAME that you define for a RACF user ID must be the same short name that is defined for that user by the administrator of the Lotus Notes for z/OS application. For special considerations when selecting application user identities, see “Considerations for Application User Names” on page 299.

Adding Application Identity Segments

The following example adds a new RACF user ID named CHEN, and associates it with an NDS user name for use with Novell Directory Services for OS/390.

```
ADDUSER CHEN NDS(UNAME('ChenMeiLing'))
```

ADDUSER command processing creates a new user profile named CHEN, adds an NDS segment to the user profile, and sets the UNAME field of the segment to ChenMeiLing.

Modifying USER Identity Segments

The following example adds an LNOTES segment to the existing user profile CHEN, and associates the RACF user ID with a short name for Lotus Notes for z/OS.

```
ALTUSER CHEN LNOTES(SNAME('ChenMeiLing'))
```

ALTUSER command processing adds an LNOTES segment to the USER profile CHEN and sets the SNAME field of the segment to ChenMeiLing.

Removing User Identity Segments

The ALTUSER command can be issued to remove the association between a RACF user ID and an application user identity. The following example deletes the NDS segment from the RACF user profile named CHEN, and removes the user ID’s association with the NDS user name ChenMeiLing.

```
ALTUSER CHEN NONDS
```

System Considerations

If your installation shares the RACF database with systems running releases prior to OS/390 Version 2 Release 10, your RACF support of Lotus Notes for z/OS and Novell Directory Services for OS/390 (NDS) is implemented using mapping profiles in the NOTELINK and NDSLINK classes. See “Mapping Profiles in the NOTELINK and NDSLINK Classes” on page 297.

If your installation shares the RACF database with only systems running z/OS, or OS/390 Version 2 Release 10 or above, you may or may not be using mapping profiles in the NOTELINK and NDSLINK class. You should see your system programmer to find out if your installation has been converted for stage 3 of application identity mapping. Stage 3 of application identity mapping uses an alias index that is automatically maintained by RACF to map application user identities, such as Lotus short names and NDS user names, without using mapping profiles in

the NOTELINK and NDSLINK classes. Once at stage 3, you can deactivate the NOTELINK and NDSLINK classes. See *z/OS SecureWay Security Server RACF System Programmer's Guide* for information about running the IRRIRA00 conversion utility to convert to stage 3 of application identity mapping.

If your installation is new to RACF and you are not running any releases prior to OS/390 Version 2 Release 10, your system will automatically use application identity mapping at the stage 3 level without running the IRRIRA00 conversion utility, and there will be no mapping profiles in the NOTELINK or NDSLINK classes.

Mapping Profiles in the NOTELINK and NDSLINK Classes

If your installation shares the RACF database with systems running releases prior to OS/390 Version 2 Release 10, or your installation shares the RACF database with only systems running z/OS, or OS/390 Version 2 Release 10 or above, but has not been converted to stage 3 of application identity mapping, your RACF support of Lotus Notes for z/OS and Novell Directory Services for OS/390 may use mapping profiles.

Mapping profiles are automatically maintained through ADDUSER, ALTUSER and DELUSER command processing when NDS and LNOTES options are specified. Each mapping profile associates a RACF user ID with an application user identity, based on the information specified in the LNOTES and NDS segments of the user profile.

The profile name for mapping profiles in the NOTELINK class is the Lotus Notes for z/OS short name (SNAME). The profile name for mapping profiles in the NDSLINK class is the Novell Directory Services for OS/390 user name (UNAME). The APPLDATA field of each mapping profile contains the RACF user ID that corresponds to the application user identity. Each application identity segment of the user profile contains one user identity name. Note that when RACF creates a mapping profile as a result of an ADDUSER or ALTUSER command, the user ID of the command issuer becomes the owner of the profile.

The following examples illustrate how mapping profiles are automatically managed by RACF.

1. A mapping profile named `ChenMeiLing` is added in the NDSLINK class, with user ID `CHEN` in the APPLDATA field, as a result of executing the following command:

   ```
   ADDUSER CHEN NDS(UNAME('ChenMeiLing'))
   ```

2. A mapping profile named `ChenMeiLing` is added in the NOTELINK class, with user ID `CHEN` in the APPLDATA field, as a result of executing the following command:

   ```
   ALTUSER CHEN LNOTES(SNAME('ChenMeiLing'))
   ```

3. The mapping profile named `ChenMeiLing` is deleted from the NDSLINK class as a result of executing the following command:

   ```
   ALTUSER CHEN NONDS
   ```

   When ALTUSER command processing removes application identity segments from user profiles, it deletes the corresponding mapping profiles in the appropriate general resource class. Using the DELUSER command to delete a user profile that contains application identity segments will also delete the corresponding mapping profiles.

Attention

If your installation uses mapping profiles, do not execute the DELUSER command for a user profile that contains identity segments from RACF systems that do not support identity mapping profiles. These systems do not automatically manage mapping profiles. You will inadvertently leave residual mapping profiles in a general resource class when the user profile is deleted. See information about recovery procedures in *z/OS SecureWay Security Server RACF System Programmer's Guide*.

In general, you should not administer mapping profiles using the RDEFINE, RALTER, RDELETE or RLIST commands. For information on correcting mapping profiles that are inadvertently deleted or damaged, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Authorizing Applications to Use Identity Mapping

Applications that do not run in system key or in supervisor state require RACF authorization to be able to use the identity mapping service (IRRSIM00). Applications that run in system key or in supervisor state do not require RACF authorization.

To authorize Novell Directory Services for OS/390 (NDS) and Lotus Notes for z/OS applications, which do not run in system key or supervisor state, you must define RACF user IDs for the applications. The RACF user IDs must be given READ access to a RACF general resource called IRR.RUSERMAP in the FACILITY class.

Defining Applications as RACF Users

Each NDS and Lotus Notes for z/OS server must be defined as a RACF user, if not already defined. It may run as a job or a started procedure.

The following example shows RACF user IDs (`LOTUS09` and `NDS14`, respectively) being defined for a Lotus Notes for z/OS server and a Novell Directory Services for OS/390 server. The user IDs are members of a RACF user group called `MAPGRP`, and the owner for all profiles is `MAPADM`.

```
ADDGROUP MAPGRP  OWNER(MAPADM)
ADDUSER  LOTUS09 GROUP(MAPGRP) OWNER(MAPADM)
ADDUSER  NDS14   GROUP(MAPGRP) OWNER(MAPADM)
```

If the application server executes as a batch job, the RACF user ID that is added is the user ID associated with the batch job. If the server executes as a started procedure, you must assign a RACF user ID using one of the following methods:

- Add the procedure name as an entry in the STARTED class. (This is the preferred method.)
- Add the procedure name in the RACF started procedure table (ICHRIN03), unless this table has already been modified by your installation to contain a generic entry.

In addition, you should assign the PROTECTED attribute to the user IDs that you associate with application servers. For more information, see "Assigning RACF User IDs to Started Procedures" on page 143.

Permitting Access to the IRR.RUSERMAP Resource

Authorization to use the identity mapping service (IRRSIM00) is controlled through a RACF general resource called IRR.RUSERMAP in the FACILITY class. You must define a profile to protect this resource and permit application user IDs to access the resource with READ authority.

Attention

Make sure an existing generic profile in the FACILITY class does not inadvertently grant this authority by default. It is recommended that you create a profile to protect the IRR.RUSERMAP resource with UACC(NONE) until you determine which applications require identity mapping.

The following example protects the IRR.RUSERMAP resource in the FACILITY class with UACC(NONE) and authorizes the group of application servers called MAPGRP to use identity mapping.

```
RDEFINE FACILITY IRR.RUSERMAP UACC(NONE)
PERMIT  IRR.RUSERMAP CLASS(FACILITY) ID(MAPGRP) ACCESS(READ)
```

Activating Identity Mapping

The FACILITY class must be active to enable identity mapping. If it is not already active at your installation, you must activate the FACILITY class using the SETROPTS command.

```
SETROPTS CLASSACT(FACILITY)
```

If your installation maintains FACILITY class profiles in storage through SETROPTS RACLIST processing, you must issue the following command to refresh the FACILITY class after you define or alter any profiles protecting the IRR.RUSERMAP resource:

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

Considerations for Application User Names

Certain application user identities, such as the Lotus Notes for z/OS short name (SNAME) and the Novell Directory Services for OS/390 (NDS) user name (UNAME), may contain blanks or may contain lower case letters.

Blanks are not permitted as part of a RACF profile name. Therefore, when building the profile name, ADDUSER and ALTUSER command processing will replace blanks with the X'4A' character (which often resolves to the ¢ symbol). RACF command processing also prevents the X'4A' character (¢) from being specified as part of an actual application user name.

You should use caution when specifying lower case letters in application user names if:

- Your installation shares the RACF database among systems that support identity mapping and systems that do not.
- Administrators may issue DELUSER commands from systems that do not support identity mapping.

  Residual mapping profiles may be left if DELUSER commands are issued from systems that do not support identity mapping. If the profile name contains blanks or lower case letters, you may be unable to remove these profiles using RACF commands. See information about recovery procedures in *z/OS SecureWay Security Server RACF System Programmer's Guide*.

General Resources

For information about the set of characters supported for application user names, see *z/OS SecureWay Security Server RACF Command Language Reference*.

Controlling Applications That Execute RACF Commands

Authorized applications, such as servers, can invoke the `R_admin` callable service (IRRSEQ00) to execute RACF commands. For detailed information about invoking the `R_admin` callable service, see *z/OS SecureWay Security Server RACF Callable Services*.

Applications that run in system key or in supervisor state do not require RACF authorization to use the `R_admin` callable service. Applications that do not run in system key or in supervisor state require RACF authorization.

To authorize applications that do not run in system key or supervisor state, you must define RACF user IDs for them. These RACF user IDs must be given READ access to RACF general resources in the FACILITY class. These resource names correspond to RACF command names and follow the form:

`IRR.RADMIN.`*`command-name`*

Defining Applications as RACF Users

Each application server must be defined as a RACF user, if not already defined. It may run as a job or a started procedure. If the application server executes as a batch job, the RACF user ID that is added is the user ID associated with the batch job. If the server executes as a started procedure, you must assign a RACF user ID using one of the following methods:

- Add the procedure name as an entry in the STARTED class. (This is the preferred method.)
- Add the procedure name in the RACF started procedure table (ICHRIN03), unless this table has already been modified by your installation to contain a generic entry.

In addition, you should assign the PROTECTED attribute to the user IDs that you associate with application servers. For more information, see “Assigning RACF User IDs to Started Procedures” on page 143.

Permitting Access to IRR.RADMIN Resources

Authorization to use the `R_admin` callable service (IRRSEQ00) is controlled through general resources called IRR.RADMIN.*command-name* in the FACILITY class. You must define a profile for each command you wish to protect, and then permit application user IDs to access the appropriate resources with READ authority. Each resource name that you define for a specific command must include the full command name even if the applications you authorize use the abbreviated form of the command name.

Attention

Make sure an existing generic profile in the FACILITY class does not inadvertently grant `R_admin` authority by default. It is recommended that you define a profile named IRR.RADMIN.* with UACC(NONE) to protect all RACF commands until you determine which commands you wish to protect and which applications you wish to authorize.

The following example protects all RACF commands using the IRR.RADMIN.* resource in the FACILITY class with UACC(NONE). It also authorizes an application server called APPL04 to use the R_admin callable service to execute the LISTUSER command.

```
SETROPTS GENERIC(FACILITY)
RDEFINE  FACILITY IRR.RADMIN.*        UACC(NONE)
RDEFINE  FACILITY IRR.RADMIN.LISTUSER UACC(NONE)
PERMIT   IRR.RADMIN.LISTUSER CLASS(FACILITY) ID(APPL04) ACCESS(READ)
```

The FACILITY class must be active to enable applications to use R_admin. If it is not already active at your installation, you must activate the FACILITY class using the SETROPTS command.

```
SETROPTS CLASSACT(FACILITY)
```

If your installation maintains FACILITY class profiles in storage through SETROPTS RACLIST processing, you must issue the following command to refresh the FACILITY class after you define or alter any profiles protecting IRR.RADMIN resources:

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

Chapter 8. Operating Considerations

This chapter discusses operating RACF on your system.

Coordinating Profile Updates

You should plan to update profiles so that they remain consistent with other profiles on the database while making sure that the updating process does not interfere with other jobs running in the system.

When RACF is enabled for sysplex communication, members of a data sharing group are notified to create, refresh, or delete their in-storage profiles. The command is coordinated to ensure that all systems begin to use the refreshed profiles simultaneously. See *z/OS SecureWay Security Server RACF Command Language Reference* for more information on the operands you need for this.

Each individual operation performed by RACF serializes on a RACF database, but a command or function can perform multiple operations on multiple profiles. For example, the CONNECT command changes both the user profile and the group profile. If two or more RACF commands or functions are executing at the same time

and are making contradictory updates, their operations might be interleaved and, therefore, cause the information in the RACF database to become incomplete or invalid.

Note: If a user is logged on, and you update the user's attributes in the RACF database using ALTUSER or CONNECT, some changes may not take effect until the next time the user enters the system. However, a LISTUSER or LISTGRP command issued immediately after the change shows the new values.

Some of the changes that are delayed until the user logs on again are the SPECIAL, OPERATIONS, and AUDITOR attributes and the list of connected groups examined by RACROUTE REQUEST=FASTAUTH.

Example:

In this example, the security administrator inadvertently creates a situation where a profile exists, but it does not have an owner. The security administrator issues DELUSER to delete a user from RACF. At the same time, the other user (who has the ADSP attribute and is logged on) creates a permanent user data set, which automatically creates a discrete data set profile.

The DELUSER command performs the following operations on the RACF database:

1. Locates the user profile in the RACF database.
2. Locates any user data set profiles.
3. Ensures that the user does not have any user data sets whose high-level qualifier is his user ID. (RACF cannot delete the user profile until all of his user data sets are deleted.)
4. Deletes the user profile.
5. Updates the group profile to remove the user as an eligible member of the group.

As a result of the ADSP attribute, RACF performs one operation on the RACF database: it adds a data set profile for the permanent user data set.

In this example, if the user adds the new data set profile between Steps 2 and 3 of the DELUSER command processing, RACF adds a user data set profile to the RACF database. However, RACF has already deleted the user who owns the profile. This creates an ownerless profile.

To prevent the creation of ownerless profiles, do not delete a user who is logged on. Instead, make sure the user is logged off and cannot log on again. If necessary, have the operator force the user off the system first. Then follow the steps described in "Summary of Steps for Deleting Users" on page 91.

RACF Commands for Flushing a VLF Cache

For installations using the IRRACEE class to store security environments with the Virtual Lookaside Facility (VLF), administrators should be aware that issuing certain RACF commands can delete one or more such objects.

Examples of commands that delete the stored security environment for a user are DELUSER, PASSWORD, and ALTUSER.

You can determine the fields that cause VLF purging on ACEE by referring to the RACF database templates in *z/OS SecureWay Security Server RACF Macros and Interfaces*. A security-sensitive field has bit 0 of flag 2 turned on. Changes to such a field trigger VLF purging.

In an installation where no RACF database sharing occurs, issuing commands that deal with certain general resource classes or profiles can delete *all* stored security environments. Examples of this include activating, deactivating, or issuing SETROPTS NORACLIST*(classname)* or SETROPTS RACLIST*(classname)* REFRESH for these classes:

APPCPORT
APPL
CONSOLE
GTERMINL
JESINPUT
SECLABEL
TERMINAL

For participants sharing a RACF database, deleting one or more stored security environments on one system causes *all* stored security environments to be deleted by the other participants. Thus, the administration of user profiles in a shared environment with a performance-oriented participant should be administered from that system, if possible.

In all cases, any deleted security environment can be restored on demand through actions such as legitimate logging on or job submission.

For information on using VLF for mapping z/OS UNIX user identifiers (UIDs) and z/OS UNIX group identifiers (GIDs) in the UNIXMAP class, see “Using the UNIXMAP Class and Virtual Lookaside Facility (VLF)” on page 508.

For more information on VLF, see *z/OS MVS Planning: Operations*, *z/OS MVS Initialization and Tuning Guide*, and *z/OS MVS Initialization and Tuning Reference*.

Getting Started with RACF (after First Installing RACF)

After you initialize your system with RACF active for the first time, you can quickly achieve system security. During RACF installation, a basic set of profiles is created in the RACF database:

Group	Superior Group	Owner	Connected Users (Group Authority)
SYS1	–	IBMUSER	IBMUSER (JOIN)
VSAMDSET	SYS1	IBMUSER	IBMUSER (JOIN)
SYSCTLG	SYS1	IBMUSER	IBMUSER (JOIN)

There is only one user:

User	Default Group (Group Authority)	Attributes	Connected Groups (Group Authority)
IBMUSER	SYS1 (JOIN)	SPECIAL and OPERATIONS	SYSCTLG (JOIN) VSAMDSET (JOIN)

And there are three SECLABELS:

Name	Owner	UACC
SYSHIGH	IBMUSER	NONE
SYSLOW	IBMUSER	NONE
SYSNONE	IBMUSER	NONE

Note: The basic set of profiles shown here is supplied with RACF. These profiles can not be defined by your installation and should not be deleted. They must exist at initialization time or RACF initialization will automatically add them.

Logging On as IBMUSER and Checking Initial Conditions

IBMUSER is the first user ID that the security administrator can use. This user ID has the SPECIAL attribute, which allows IBMUSER to issue most of the RACF commands (except for those reserved for users with the AUDITOR attribute) and the OPERATIONS attribute, which allows IBMUSER to access many RACF-protected resources.

When you enter the system for the first time with the IBMUSER user ID, you must change the initial password, SYS1, to a new password. A new password prevents any other user from entering the system as IBMUSER.

Log on as IBMUSER:

```
LOGON IBMUSER
```

After entering IBMUSER's old password (SYS1) and defining a new password, list the system-wide RACF options that are in effect:

```
SETROPTS LIST
```

Read through this list to familiarize yourself with the options that are in effect. For an explanation of what some of the options are and what they mean, see "Using the SETROPTS Command" on page 112.

Note: Not all options are displayed at this point, because IBMUSER does not have the AUDITOR attribute. If you want to see the status of these options, grant IBMUSER the AUDITOR attribute, log off, and log on again. To see all of the options, issue SETROPTS LIST again.

For a complete listing of all of the options that are available, see *z/OS SecureWay Security Server RACF Command Language Reference*.

Attention

The option for the TERMINAL resource class should be specified as READ. Do not change it to NONE unless you have defined your terminals to RACF and authorized the appropriate users and groups to access them. If you specify TERMINAL(NONE) without first defining your terminals to RACF, you cannot access your terminals and, consequently, you will be locked out of your system.

Defining Administrator User IDs for Your Own Use

Define a new user (for example, RACFADM) to RACF for your own use. This user should have at least the SPECIAL and OPERATIONS attributes. If you are also the system-wide auditor, you should also give this user ID the AUDITOR attribute. Depending on the attributes you select, enter one of the following commands:

- `ADDUSER RACFADM SPECIAL OPERATIONS`
- `ADDUSER RACFADM SPECIAL OPERATIONS AUDITOR`

Note: You should also plan this user's SYS1.UADS entry (see *z/OS TSO/E Customization*) or TSO segment. For example, if an administrator (including IBMUSER) is to work with data set profiles (using the ADDSD or ALTDSD commands), you should ensure that the user can have volumes mounted during a TSO session. You can do this by giving the user the MOUNT attribute in the SYS1.UADS entry, or by giving the user READ access authority to the MOUNT profile in the TSOAUTH class. See "Protecting TSO Resources" on page 496.

Defining at Least One User ID to Be Used for Emergencies Only

To handle emergency situations that could arise, such as if RACF becomes inoperative or all SPECIAL users become revoked, you should consider setting up at least one, and preferably two, "emergency" user IDs. These user IDs should *never* be used except in extreme cases, under management supervision. They should have no TSO segment in their RACF user profiles, and their entries in the SYS1.UADS data set should give them all attributes.

Logging on as RACFADM, Checking Groups and Users, and Revoking IBMUSER

Log on as RACFADM and use the default password, SYS1 in this case (IBMUSER's default group).

First, list all users to ensure that only RACFADM and IBMUSER are defined to RACF, and that they have the proper attributes.

At this point, you receive a message stating that your password expired. Immediately change the password, SYS1, to a new password.

```
LISTUSER *
```

Then, list all of the groups that are defined to RACF:

```
LISTGRP *
```

Connect RACFADM to them and make RACFADM the owner of the groups:

```
CONNECT RACFADM GROUP(SYS1) AUTH(JOIN)
CONNECT RACFADM GROUP(SYSCTLG) AUTH(JOIN)
CONNECT RACFADM GROUP(VSAMDSET) AUTH(JOIN)

ALTGROUP SYS1 OWNER(RACFADM)
ALTGROUP SYSCTLG OWNER(RACFADM)
ALTGROUP VSAMDSET OWNER(RACFADM)
```

Then, revoke the IBMUSER user ID so that another user cannot use it:

```
ALTUSER IBMUSER REVOKE
```

Note: You cannot delete the IBMUSER user profile.

Operating Considerations

Define another user to RACF (for example, user ID RACFAD2), to act as your assistant. Make the new user's default group SYS1, and give this assistant the SPECIAL and OPERATIONS user attributes.

```
ADDUSER RACFAD2 DFLTGRP(SYS1) AUTH(JOIN) SPECIAL OPERATIONS
```

Defining the Groups Needed for the First Users

At this point you should consider creating the groups that you need.

The following commands show an example of adding four groups. Three are departmental groups (GROUP1, GROUP2, and GROUP3), and GROUP2 and GROUP3 are owned by GROUP1 so that certain authorities can be propagated. The fourth group (DATAMGT) has global pack maintenance responsibility.

```
ADDGROUP (GROUP1 DATAMGT)
ADDGROUP (GROUP2 GROUP3) OWNER(GROUP1) SUPGROUP(GROUP1)
```

Defining a System-Wide Auditor

Define a user (for example, AUDCCC) who has system-wide auditing responsibilities and privileges.

```
ADDUSER AUDCCC AUDITOR
```

Defining Users and Groups

You now add a user (D03DIK) to GROUP3 with authority to protect group data sets.

```
ADDUSER D03DIK OWNER(GROUP3) AUTH(CREATE) DFLTGRP(GROUP3)
```

Note: For more information, see "Summary of Steps for Defining Users" on page 88.

Defining Group Administrators, Group Auditors, and Data Managers

For each group, define a group administrator with the group-SPECIAL attribute. Only the administrator for GROUP1 has the authority to define new users in that group. Each of the other administrators has authority over the resources owned by his or her group, as well as the resources owned by users who are owned by his or her group.

```
ADDUSER D01RHG DFLTGRP(GROUP1) CLAUTH(USER) DATA('GROUP1 ADM')
CONNECT D01RHG GROUP(GROUP1) AUTH(JOIN) SPECIAL

ADDUSER D02JMP DFLTGRP(GROUP2) DATA('GROUP2 ADM')
CONNECT D02JMP GROUP(GROUP2) AUTH(CREATE) SPECIAL

ADDUSER D03ABL DFLTGRP(GROUP3) DATA('GROUP3 ADM')
CONNECT D03ABL GROUP(GROUP3) AUTH(CREATE) SPECIAL
```

For groups GROUP1, GROUP2, and GROUP3, define a group-auditor. Connect the user to GROUP1 and give the user the group-AUDITOR attribute. Because GROUP2 and GROUP3 are owned by GROUP1, the user has auditor authority over the resources and users belonging to those groups, as well as to GROUP1. The user does not have auditor authority in any other group.

```
ADDUSER D01GPB DFLTGRP(GROUP1) DATA('AUDITOR G1 G2 G3')
CONNECT D01GPB GROUP(GROUP1) AUDITOR
```

The administrator for the data management group, the data manager, is able to define DASD volumes to RACF in order to perform dump, restore, and data cleanup operations.

```
ADDUSER DMGJFS DFLTGRP(DATAMGT) AUTH(JOIN) CLAUTH(USER DASDVOL)
        DATA('DATA MGT ADM')
```

Because of his or her duties, the data manager is connected to SYS1, allowing the manager to access data sets with SYS1 in their access list and to define SYS1 data set profiles to RACF. The data manager has the group-SPECIAL attribute in group SYS1.

```
CONNECT DMGJFS GROUP(SYS1) AUTH(CREATE) UACC(READ) SPECIAL
```

At the end of the session, the defined group structure is:

Group	Superior Group	Owner	Connected Users (Group Authority)
SYS1	–	RACFADM	IBMUSER (JOIN) RACFADM (JOIN) RACFAD2 (JOIN) DMGJFS (CREATE)
VSAMDSET	SYS1	RACFADM	IBMUSER (JOIN) RACFADM (JOIN)
SYSCTLG	SYS1	RACFADM	IBMUSER (JOIN) RACFADM (JOIN)
GROUP1	SYS1	RACFADM	D01RHG (JOIN) D01GPB (USE)
GROUP2	SYS1	GROUP1	D02JMP (CREATE)
GROUP3	SYS1	GROUP1	D03ABL (CREATE)
DATAMGT	SYS1	RACFADM	DMGJFS (JOIN)

The defined users are:

User	Default Group (Group Authority)	Attributes	Connected Groups (Group Authority)
IBMUSER	SYS1 (JOIN)	SPECIAL, OPERATIONS, REVOKE	SYSCTLG (JOIN), VSAMDSET (JOIN)
RACFADM	SYS1 (JOIN)	SPECIAL, AUDITOR, OPERATIONS	SYSCTLG (JOIN), VSAMDSET (JOIN)
RACFAD2	SYS1 (JOIN)	SPECIAL, OPERATIONS	
DMGJFS	DATAMGT (JOIN), SYS1(CREATE)	CLAUTH(USER DASDVOL), SPECIAL	SYS1(CREATE)
D01RHG	GROUP1 (JOIN)	CLAUTH(USER), group-SPECIAL	
D02JMP	GROUP2 (USE)	group-SPECIAL	
D03ABL	GROUP3 (CREATE)	group-SPECIAL	
D01GPB	GROUP1 (CREATE)	group-AUDITOR	
D03DIK	GROUP3 (CREATE)		
AUDCCC	SYS1 (USE)	AUDITOR	

Protecting System Data Sets

Create data set profiles to protect your system data sets. These should include the data sets described in Table 60 on page 625, as well as other data sets that your installation considers sensitive. The following are some candidates:

- General installation libraries
  - PROCLIB
  - TSO help and CLISTs
  - Compiler libraries
  - SORTLIB
- System control libraries
  - Nucleus, SVCLIB, LPALIB
  - Spool and paging data sets
  - APF-authorized libraries
  - Master catalog
  - DLIB data
  - SYS1.UADS
  - PARMLIB
- Sensitive data
  - Corporate trade secrets
  - Research results
  - Employee data
  - Customer or client lists
- Production libraries
  - PROCLIBs
  - LOADLIBs
- Application development programs and data
  - Source
  - Load libraries
  - Documentation
- User data
  - JCL
  - Documentation
  - Source
  - Load modules

Setting RACF Options

Review “Chapter 5. Specifying RACF Options” on page 111 for the RACF options you wish to set:

- Selecting options with the SETROPTS command (see “Using the SETROPTS Command” on page 112)
- Encrypting RACF user passwords (see “Specifying the Encryption Method for User Passwords” on page 142)
- Using started procedures (see “Using Started Procedures” on page 143)

Using the Data Security Monitor (DSMON)

The data security monitor (DSMON) produces a set of reports that provide information about the current status of the data security environment at your installation.

The reports DSMON can produce are:

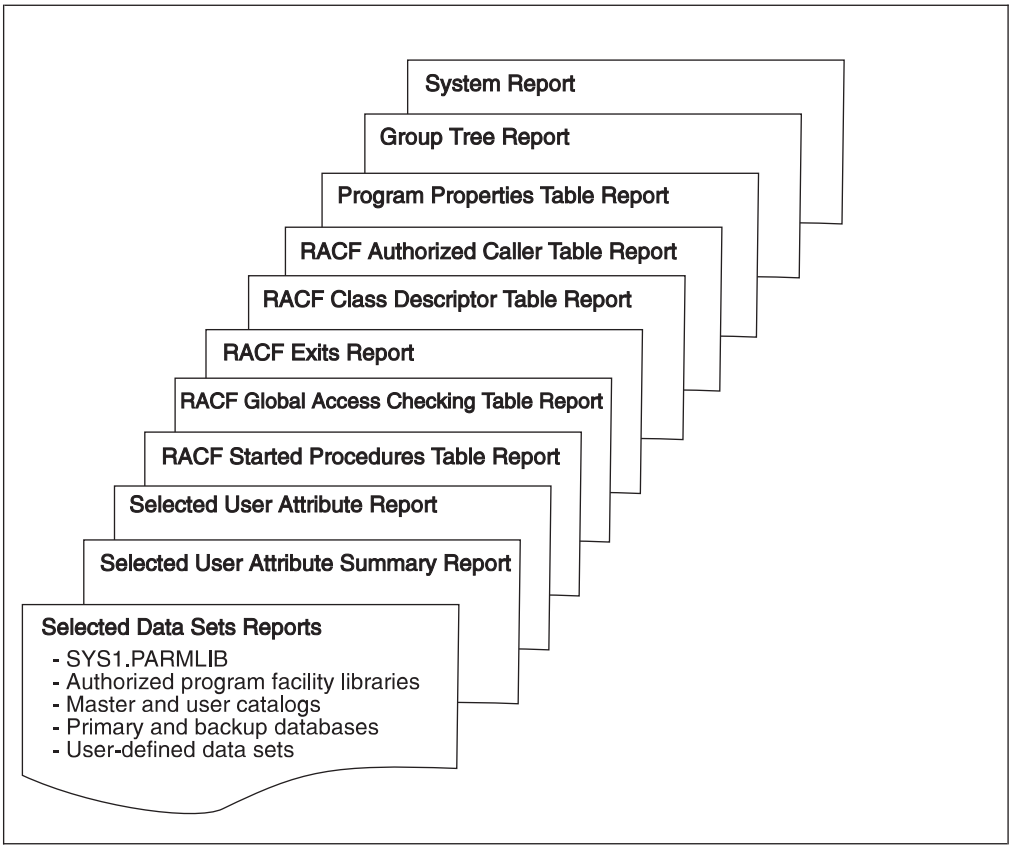


Figure 15. Reports produced by DSMON

These reports can help you (1) check the initial steps you took to establish system security, and (2) make additional security checks periodically.

A short description of each report follows. See *z/OS SecureWay Security Server RACF Auditor's Guide* for more information on these reports and how to invoke the data security monitor.

The System Report

The system report contains information such as the identification and model of the processor complex, and the name, version, and release of the operating system. This report also specifies the RACF version and release number and whether RACF is active. If RACF is inactive, DSMON prints a message that tells you whether RACF was not activated at IPL or was deactivated by the RVARY command.

The Group Tree Report

This report lists, for each requested group, all of its subgroups, all of the subgroups' subgroups, and so on, as well as the owner of each group listed in the report, if the owner is not the superior group.

You can use the group tree report to examine the overall RACF group structure for your system. You can also determine the scope of the group for group-related user attributes (group-SPECIAL, group-OPERATIONS, and group-AUDITOR).

The Program Properties Table Report

This report lists all of the programs in the MVS program properties table

(PPT). The report also indicates, for each program, whether the program is authorized to bypass password protection and whether it runs in a system key.

You can use the program properties table report to verify that only those programs that the installation has authorized to bypass password protection are, in fact, able to do so. Such programs are normally communication and database control programs, and other system control programs.

You can also verify that only those programs that the installation has authorized are able to run in a system key.

The RACF Authorized Caller Table Report

This report lists the names of all of the programs in the RACF authorized-caller table. The programs in this table are authorized to issue the RACROUTE REQUEST=VERIFY macro to perform user verification, or the RACROUTE REQUEST=LIST macro to load profiles into main storage.

You can use this report to verify that only those programs that are supposed to be authorized to modify an ACEE (accessor environment element) are able to issue the RACROUTE REQUEST=VERIFY. This verification is a particularly important security requirement because the ACEE contains a description of the current user. This description includes the user ID, the current connect group, the user attributes, and the group authorities. A program that is authorized to issue the RACROUTE REQUEST=VERIFY could alter the ACEE to simulate any user.

You can also use this report to verify that only those programs that are supposed to be authorized to access profiles are able to issue the RACROUTE REQUEST=LIST. Because profiles contain complete descriptions of the characteristics that are associated with RACF-defined entities, you must carefully control access to them.

The RACF Class Descriptor Table Report

This report lists, for each general resource class in the class descriptor table (CDT), the class name, the default UACC, whether the class is active, whether auditing is being done, whether statistics are being kept, and whether OPERATIONS attribute users have access.

You can use the class descriptor table report to determine which classes (besides DATASET) are defined to RACF and active, and therefore can contain resources that RACF protects.

The RACF Exits Report

This report lists the names of all of the installation-defined RACF exit routines and specifies the size of each exit routine module.

You can use the RACF exits report to verify that the only active exit routines are those that your installation has defined. The existence of any other exit routines might indicate a system security exposure, because RACF exit routines can be used to bypass RACF security checking. Similarly, if the length of an exit routine module differs from the length of the module when it was defined by your installation, the module might have unauthorized modifications.

The RACF Global Access Checking Table Report

This report lists, for each resource class in the global access table, all of the entry names and their associated resource access authorities.

Because global access checking allows anyone to access the resource at the associated access authority, you should verify that each entry has an appropriate level of access authority.

The RACF Started Procedures Table Reports

RACF generates two reports about the started procedures table (ICHRIN03).

- If the STARTED class is active, the report uses the STARTED class profiles and contains the TRACE attribute. The trace uses module ICHDSM00.
- If the STARTED class is not active, the trace uses the installation replaceable load module, ICHRIN03.

The reports list the procedure name, the user ID and group name to be associated with the procedure, and whether the procedure is privileged or trusted.

You can use the report to determine which started procedures are defined to RACF, and which have the privileged attribute. If a started procedure is privileged or trusted, it bypasses all REQUEST=AUTH processing (unless the CSA or PRIVATE operand was specified on REQUEST=AUTH), including checks for security classification of users and data.

The Selected User Attribute Report

The selected user attribute report:

- Lists all RACF users with the SPECIAL, OPERATIONS, AUDITOR, or REVOKE attributes
- Specifies whether they possess these attributes on a system-wide (user) or group level
- Indicates whether they have any user ID associations

You can use this report to verify that only those users who need to be authorized to perform certain functions have been assigned the corresponding attribute.

Selected User Attribute Summary Report

The selected user attribute summary report shows the number of installation-defined users and totals for users with the SPECIAL, OPERATIONS, AUDITOR, and REVOKE attributes, at both the system and group level. You can use this report to verify that the number of users with each of these attributes, on either a system or group level, is the number that your installation wants. In particular, you should make sure that you have assigned the SPECIAL attribute (on a system level) to at least one user and the AUDITOR attribute (on a system level) to at least one user.

The Selected Data Sets Report

This report lists the names of selected system data sets and, for each data set, specifies the criterion for selection, the serial number of the volume on which it resides, whether the data set is RACF-indicated or RACF-protected, and the universal access authority (UACC). If a data set meets more than one selection criterion, there is a separate entry in the report for each criterion. The selected data sets include system data sets, the MVS master catalog, user catalogs, the RACF primary and backup data sets, and user-specified data sets.

You can use the selected data sets report to determine which of these data sets are protected by RACF and which are not. You can also check whether

the UACC associated with each of the data sets is compatible with your installation's resource access control requirements.

JCL Parameters Related to RACF

This section summarizes the JCL parameters that relate to RACF. For complete information, see *z/OS MVS JCL Reference*.

- On the JOB statement:
  - USER parameter: Specify this parameter if user ID propagation is not used or if the user is submitting a job for another user.
  - PASSWORD parameter: Specify this parameter *only* when absolutely necessary. Specifying this parameter in JCL exposes the password to potential misuse.

    Note: If a JOB statement contains a RACF password, you should establish procedures to ensure the security of the JOB statement. For example, if the passwords are in a card deck, users can put passwords on a continuation statement and use non-printing punches so that the passwords are not easily visible.

    JES suppresses the printing of passwords in output listings.
  - GROUP parameter: Specify this parameter only if list-of-groups processing is *not* in effect and if the user wants the job to run with a group other than the user's default group.
  - SECLABEL parameter: Specify this parameter if the job is to run with a security label other than the user's current security label.

    If user ID propagation is used, all of these parameters are optional. Also, a TSO SUBMIT installation exit, TSO, or other procedures for handling batch jobs can place the RACF parameters on the JOB statement.
- On the DD statement:
  - PROTECT parameter.
  - LABEL parameter.
  - MGMTCLAS parameter.
  - STORCLAS parameter
  - DSNAME parameter: Use the DSNAME parameter to assign a temporary data set name to an in-stream data set and to a SYSOUT data set. This name can be specified as a qualifier in JESSPOOL profile names. For more information, see "Defining Profiles for SYSIN and SYSOUT Data Sets" on page 471.

    When creating new data sets or tape volumes that require a new discrete profile, specify PROTECT=YES to automatically define the discrete profile.

    Note: If the data set being created is adequately covered by a generic profile, do not use the PROTECT parameter because this forces the creation of a discrete profile.
  - SECMODEL parameter: When creating new data sets or tape volumes that require a new discrete profile, specify the SECMODEL parameter to copy an existing data set profile to the new discrete data set profile.

    Note: If the data set being created is adequately covered by a generic profile, do not use the SECMODEL parameter because this forces the creation of a discrete profile.
- On the OUTPUT statement:

- DPAGELBL parameter: With PSF for OS/390 is installed, use the DPAGELBL parameter to indicate whether the system should print information related to the job's security label on each page of printed output.
- SYSAREA parameter: use the SYSAREA parameter to indicate whether the system should reserve an area on each page of printed output for information related to the security label.

Restarting Jobs

When a job automatically restarts and returns to a previous checkpoint, RACF repeats user verification and access authorization checking. If the job changed the password on the JOB statement, RACF uses the new password for user verification. But meanwhile, if the PASSWORD command or another job changes the password, RACF detects an invalid password and fails the job.

When you submit a job for a deferred restart, you can specify your current password on the JOB statement, or use JES user ID propagation and avoid the problem of exposing your password on the JOB statement.

For either an automatic or deferred restart, the user's current access authority is checked (the access authority at the time of the restart), and is used for all resources the job tries to access.

Bypassing Password Protection

You can authorize certain programs to bypass password protection on resources such as data sets and volumes. You do this by specifying a setting in the MVS program properties table. Programs that you would normally authorize to bypass password protection include communication and database control programs, and other system control programs.

RACF can be used to perform authorization checking, but RACF itself does not check the bypass password protection setting. Instead, programs that use RACF services, such as DFSMS, check the setting to determine whether to call RACF. Therefore, for information about how a product uses the bypass password protection setting, see its documentation.

Controlling Access to RACF Passwords

Installation personnel should ensure that the security of RACF user passwords is not violated.

You should restrict the operator's use of the JES operator commands. Using JES commands, the system operator can display JES data areas that contain both the current and new RACF passwords associated with a job, even though these passwords are in a masked format. (When a user submits a job and supplies RACF passwords on the JOB statement, JES stores them for the life of the job.)

It is also possible for the operator to display password information when displaying real storage at the console. Again, the installation should monitor the operator's activities to ensure that passwords remain secure. You can monitor the operator by logging the operator commands (with the SETROPTS OPERAUDIT command) or by creating profiles for the appropriate commands in the OPERCMDS class, and requesting auditing in the OPERCMDS profiles.

Also, the JES3 dump core utility allows users to view stored passwords. You should restrict access to the JES3 dump core utility.

Note: JES3 allows system programmers to specify a password for the JES3 dump core utility (not a RACF password). This password is stored in clear text in a JES3 module. You should protect this module from unauthorized use.

RACF commands that contain passwords should not be issued on the operator's console because these passwords would then show up in SYSLOG. Also, make sure you protect the GTF trace dataset if you have SET TRACE active because passwords might appear in the trace records that are produced.

You should restrict access to SVC dumps and stand-alone dumps, which might contain password information.

If users need to submit jobs for other users, activate the SURROGAT class and define profiles so that users can allow other users to submit jobs for them. In this way, a user does not need to know anyone else's password. For more information, see "Surrogate Job Submission" on page 451.

If JES support for user ID propagation is installed, batch jobs submitted by TSO users do not need any RACF identification information (user ID, group name, and password) in their JOB statements, as long as the following assumptions are true:

- The TSO user is RACF-defined.
- The job is submitted under the user's own user ID.
- If the job is submitted on a processor that is part of an NJE (networking job entry) network, the job runs on the home (user's) node.

Note: If a user specifies //DD DATA and neglects to delimit the data (with /* or DLM specification) when submitting a batch job through a card reader or RJE work station, subsequent jobs are read as part of the user's data until a delimiter is read. You should be aware that if this situation occurs, RACF user IDs, group names, passwords, and resource names from the following job's JCL become available to the user who failed to supply a delimiter. The installation should use SMF or JES installation-written exit routines to restrict the use of the //DD DATA statement to reduce this security exposure.

Authorizing Only RACF-Defined Users to Access RACF-Protected Resources

If the universal access authority (UACC)for a RACF-protected resource is READ or higher:

- Non-RACF-defined users can access the RACF-protected resource with the specified level of universal access.
- Users who enter the system using shared RACF-defined user IDs without the RESTRICTED attribute, can access the RACF-protected resource with the specified level of universal access. These users include those who enter the system:
  1. By presenting digital certificates that are not registered to RACF, who are assigned shared user IDs based on certificate name filtering.
  2. By accessing application servers that allow users to enter the system without identifying themselves, who are assigned shared user IDs such as PUBLIC or ANONYMOS.

Note: For more information, see “Defining Restricted User IDs” on page 87.

- RACF-defined users who have access authority of NONE can access the resource with the specified level of universal access by submitting a batch job without specifying the USER operand on the JCL JOB statement.

The entry `ID(*)` can be added to the access list to ensure that only RACF-defined users (who do not have the RESTRICTED attribute) can access a protected resource. For more information, see “Using ID(*) on the Access List” on page 9.

These accesses to RACF-protected resources can be prevented using the SETROPTS BATCHALLRACF and XBMALLRACF options, or by the REQUEST=AUTH preprocessing exit routine that fails REQUEST=AUTH processing for users who have entered the system using the RACF default user ID.

If JES user ID propagation is not in effect, this REQUEST=AUTH processing requires RACF-defined users to identify themselves (using the USER operand) on batch jobs that access RACF-protected resources and prevents non-RACF users from accessing RACF-protected resources.

Using the TSO or ISPF Editor

If a user edits a RACF-protected data set to which the user has only READ access authority, a failure occurs when the user attempts to save the data set. To issue the SAVE command, the user must have at least UPDATE access authority to the data set.

Service by IBM Personnel

If IBM support personnel require access to the system for servicing, they must be defined to RACF if they need to access RACF-protected data sets for servicing. Also, they need the appropriate access authority to these data sets.

You can define user profiles for IBM support personnel with the REVOKE attribute set. Then an authorized installation user can set (and reset), as needed, the REVOKE attribute in the user profile to allow IBM support personnel to enter the system. (The REVOKE and RESUME operands of the ALTUSER or CONNECT command alter the REVOKE attribute. See *z/OS SecureWay Security Server RACF Command Language Reference* for more information.

Failsoft Processing

During failsoft processing (when the RACF database is not active), RACF uses global access checking tables, REQUEST=LIST in-storage profiles, or a supplied profile, if any of these are present, to process resource access checking requests.

Note: RACF does not perform generic profile checking, because a generic profile might allow access to a resource that an existing discrete profile already protects. If that profile had been retrieved, RACF would not have allowed access to the resource.

RACF calls REQUEST=AUTH and REQUEST=DEFINE preprocessing installation exits during failsoft processing. (RACF does not call postprocessing exits.) This action frees the installation to define its own version of failsoft processing. By defining its own version of failsoft processing, an installation can allow or deny access to a resource or permit normal failsoft processing to continue.

Operating Considerations

During failsoft processing, the logging that your installation has specified continues as when RACF is active. In addition, RACF logs all accesses that the operator allows or denies.

If no global access checking tables are present, no REQUEST=LIST in-storage profiles are present, and no profile has been supplied, the preprocessing installation exits are called first. Then failsoft processing continues as follows:

1. RACROUTE REQUEST=AUTH:
   - For started procedures, RACF issues an information message to the operator to describe the name and access mode of the resource. If the started procedure does not have the privileged attribute through the RACF started procedures table, RACF issues an operator intervention message to request permission to allow access to the resource.
   - For TSO sessions, RACF issues the information message and, if the high-level qualifier of the data set name matches the user's TSO user ID, RACF allows access to the resource. If the high-level qualifier does not match the user's TSO user ID, RACF also issues an operator intervention message to request permission to allow access to the resource. If the system operator gives a negative response to a request for access, the request is denied, with, in some cases, an ABEND.
   - For all other environments, RACF issues the information message, followed by the operator intervention message. If the system operator gives a negative response to a request for access, the request is denied.
2. REQUEST=DEFINE:

   RACF issues an operator message to indicate that REQUEST=DEFINE has been issued and that the request is allowed. If the user had the ADSP attribute, or if PROTECT=YES was specified on the JCL for the data set, the resource may be RACF-indicated without a RACF discrete profile being created.

   You can use the operator message or SMF log records at a later time to determine whether the specified resource is in the RACF database. If it is not, use the ADDSD or RDEFINE command to create a profile for the resource.

Failsoft Processing with Tape Data Sets

When RACF is maintaining TVTOCs, RACF checks the TVTOC during normal processing to determine the authority level that is required to define a data set or add a data set to a volume. In failsoft mode, RACF cannot make any of the normal consistency checks to ensure that the user is only writing to the last data set on the volume and is authorized to the current data on the tape volume.

Considerations for RACF Databases

The RACF database contains all RACF access control information. RACF databases can reside on any DASD device that is supported by the operating system. Each volume that contains a RACF database should be permanently resident. If RACF is heavily used, plan to put the database on a device accessed by a channel and control unit that is least likely to affect system performance.

Backup RACF Database

RACF allows you to provide a backup database to which you can switch should your primary RACF data base fail. A backup RACF database reflects the contents of the primary. Once the installation creates the backup, RACF can maintain it automatically.

For more information about setting up a backup RACF database, maintaining RACF databases, and switching to alternate RACF databases, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Multiple Data Set Support

As an alternative to maintaining all RACF profiles of your primary RACF database in a single MVS data set, RACF allows you to have up to 90 primary data sets, and an equal number of associated backup data sets. You should consider using multiple RACF data sets to reduce device contention and to reduce the number of resources that are made unavailable by the loss of one data set or device (although for installations running in data sharing mode, a single data set may provide satisfactory performance). The best number of RACF data sets for your installation depends on the extent to which you use RACF.

For more information about creating multiple RACF data sets, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Protecting the RACF Database

It is very important that the data sets containing the primary and backup databases are properly protected. You should also ensure that data sets containing RACF database information, such as backup copies and unloaded versions of the RACF database, are also protected. In protecting these data sets, you should ensure that only those users who have a definite job-related need to read or update the data have access. Any other users should not have access to the data sets containing your RACF databases.

- These data sets should be protected with data set profiles that specify UACC(NONE), NOWARNING, and ERASE. The profiles should not have ID(*) in the access list. The NOTIFY user ID should be the RACF security administrator. System programmers who need to use the block update command (BLKUPD) to repair the RACF database must have UPDATE authority to the database. System programmers and others who need to run IRRUT400 or IRRUT200 to copy the database will need READ authority (or UPDATE, if using the LOCKINPUT or UNLOCKINPUT parameters of IRRUT400). Anyone who needs to run IRRDBU00 against a RACF database will also need UPDATE access, but it may be better to give them READ access and have them make a copy of the database using IRRUT200, then run IRRDBU00 against the copy.
- If the installation uses profiles in the DASDVOL class to allow access to volumes, you should strictly limit the number of users who have READ access authority to the volumes that hold the data sets containing the RACF database. For more information, see "DASD Volume Authority" on page 173.

Note: If making a copy of the database for the purpose of running IRRDBU00, don't forget to protect the copy as you would the database itself, including the use of ERASE.

Using RACF Data Sharing

You can also reduce device contention to the RACF database by using RACF data sharing. You also need to consider the following when the system is enabled for sysplex communication:

- Each member of the sysplex data sharing group must share the same database.
- The members of the sysplex data sharing group cannot share the database with any non-member system.
- Each system must use a compatible data set name table (ICHRDSNT).

Operating Considerations

- Each system must use an identical database range table (ICHRRNG).

For more information about shared or multiple RACF databases, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Sharing Data without Sharing a RACF Database

You might find it useful to share RACF data between systems. The RACF remote sharing facility (RRSF) removes the restrictions of shared DASD. It allows you to configure your systems into a network of *RRSF nodes* communicating via VTAM and APPC/MVS, and share RACF data between these nodes regardless of their physical proximity. You can:

- Give each RRSF node its own copy of the same RACF database and use remote sharing functions to keep the databases synchronized.
- Synchronize subsets of the database information, such as the user profiles.
- Administer RACF databases remotely.
- Automatically synchronize passwords for specified user IDs on systems in the RRSF network.

For more information on administering the RACF remote sharing facility, see "Chapter 10. The RACF Remote Sharing Facility (RRSF)" on page 355.

Number of Resident Data Blocks

IBM highly recommends that you use resident data blocks to reduce the number of I/O requests made to the RACF database. See *z/OS SecureWay Security Server RACF System Programmer's Guide* for details.

Chapter 9. Working With The RACF Database

This chapter describes tasks related to working with and maintaining the RACF database using the database unload utility (IRRDBU00) and the remove ID utility (IRRRID00).

For information about using the SMF data unload utility (IRRADU00), see *z/OS SecureWay Security Server RACF Auditor's Guide*.

The RACF database holds an installation's security data. This data is used to control access to resources, verify users, and generate a variety of reports dealing with system usage and integrity. Standard reports are provided and used to determine whether the installation's security objectives are being met.

Using the RACF Database Unload Utility (IRRDBU00)

The RACF database unload utility enables installations to create a sequential file from a RACF database. The sequential file can be used in several ways: viewed directly, used as input for installation-written programs, and manipulated with sort/merge utilities. It can also be uploaded to a database manager, such as DB2, to process complex inquiries and create installation-tailored reports.

Note: Whenever you need to run the database unload utility against a database that is active on a system that is a member of the RACF sysplex data sharing group, always run the utility from a system in the group. If you do not, you receive unpredictable results from the utility.

Diagnosis

You can use the IRRDBU00 utility for some diagnosis functions. Because this utility reads every profile in the RACF database, it also validates such profile data as lengths and count fields that are needed to read each profile successfully.

This validation can be used to help identify a profile in error. If IRRDBU00 encounters a profile in error, it may issue message IRR67092. This message contains an ICHEINTY return and reason code and also the entry name of the profile being processed.

If a profile abends or ends in another fashion without receiving this message, you may also be able to determine the profile in error. To do this, look in the output data set (OUTDD) and find the last profile (at the bottom), that was unloaded. It is likely that this profile is okay. However, the next profile in the database (in the same class) could possibly be the profile in error, if indeed a bad profile is causing the utility to end.

The next profile in the database can be found by examining the output of an IRRUT200 utility run (specifically, INDEX FORMAT), or by using the block update command (BLKUPD) to examine an online database.

For more information on diagnosing and correcting the RACF database, see *z/OS SecureWay Security Server RACF System Programmer's Guide* and *z/OS SecureWay Security Server RACF Diagnosis Guide*.

Performance Considerations

IRRDBU00 processes either a copy of the RACF database, a backup RACF database, or the active RACF database. You must have UPDATE authority to the database. It is *recommended* that you run the utility against a recent copy of your RACF database using the NOLOCKINPUT option.

While processing, IRRDBU00 serializes on one profile at a time (this is also the case in IRRUT100 processing). When IRRDBU00 has finished copying a profile, it releases the serialization. Consider this possible impact to performance if you select your active RACF database as input. Running IRRDBU00 against a *copy* of the database causes the least impact to system performance.

Note: If your system is enabled for sysplex communication RACF serializes database accesses by using global resource serialization instead of hardware RESERVEs when the system is operating in data sharing mode or in read-only mode and unloading an active database.

An installation can choose to unload its database with one utility invocation, or if it has split its database, it can unload individual pieces of its database with separate utility invocations. These utility invocations can execute concurrently.

Operational Considerations

The output records of IRRDBU00 are determined by the structure of the RACF database. The utility unloads all of the profiles in the database. It does not unload all of the fields in each profile and treats some fields in a special way. Fields that contain customer data are unloaded exactly as they appear in the database. Encrypted and reserved fields are not unloaded. Although the maximum length unloaded for most fields is 255 bytes, all 1023 bytes of data for the HOME and PROGRAM fields in the user's OMVS segment and the HOME, PROGRAM, and FSROOT fields in the user's OVM segment are unloaded.

See *z/OS SecureWay Security Server RACF Macros and Interfaces* for the conversion rules of the database unload utility.

The database unload utility uses both the supplied class descriptor table (ICHRRCDX) and the installation-defined class descriptor table (ICHRRCDE) as it unloads profiles. If your database is imported from another system, you may also have to import ICHRRCDX and ICHRRCDE. Classes are unloaded only if there is an entry for them in ICHRRCDE or ICHRRCDX on the system running the utility.

To correlate the RACF profiles with the data unloaded by the utility, see *z/OS SecureWay Security Server RACF Macros and Interfaces*.

Using IRRDBU00 with Universal Groups

Using the output from IRRDBU00 is the best way to list the members of a universal group, because a complete member list for a universal group may not be obtained from a LISTGRP command. You can use DB2 to process the output of IRRDBU00 to generate a list of universal groups and to list the members of each universal group by using samples available in SYS1.SAMPLIB. See "Using the Database Unload Utility Output with DB2" on page 333 for more information. Sample RACFICE reports called GPRM and CUG$ are also available to assist you. See "Reports Based on the Database Unload Utility (IRRDBU00)" on page 329.

Note that IRRDBU00 does not unload a Group Members data record (0102) for every user connected to a universal group. Only users who are listed in the group's member list (users with group-level user attributes, such as group-SPECIAL, or group authority higher than USE) have 0102 records.

For more information about universal groups, see "Defining Large Groups with the UNIVERSAL Attribute" on page 54.

Running the Database Unload Utility

The following job control statements are necessary for executing IRRDBU00:

JOB Initiates the job.

EXEC Specifies the program name (PGM=IRRDBU00) or, if the job control statements are in a procedure library, the procedure name. You must request IRRDBU00 processing options by specifying a parameter in the PARM field.

SYSPRINT DD
Defines a sequential message data set.

INDD*n* DD Defines the RACF input data set that makes up the RACF database. The input data sets must have all of the characteristics of a RACF database; that is, they must be contiguous single-extent data sets, non-VIO, with a logical record length (LRECL) of 4096 and a record format (RECFM) of fixed (F).

The *n* in INDD*n* refers to the location of the data set name in the data set name table (ICHRDSNT). If you have not split your RACF database, you only have to specify INDD1. If you have split your RACF database, you can unload each part with a separate utility invocation and specify INDD1 for the input data set, or you can unload all of the parts with one utility invocation.

Note: When unloading all parts, specify INDD statements in the same order as they appear in the RACF data set name table. For example:

INDD1	First data set name
INDD2	Second data set name

See “Input Data Set Specification”.

OUTDD DD Defines the single sequential output data set. The output of IRRDBU00 is a set of variable length records.

The size of the output data set is roughly estimated as twice the size of the used portion of the input data set, but you must also consider the type of profiles in your database. For example, profiles that have variable length fields, such as installation data, require more space when they are unloaded, because the maximum size of the field is unloaded (up to 255 bytes or, for the HOME and PROGRAM fields, up to 1023 bytes).

Determine the percentage of space your database is using by running the IRRUT200 utility, and use that percentage to guide you in allocating the output file. For example, if your database has 100 cylinders allocated and you are using 35% of it, you need approximately 70 cylinders for your output file.

OUTDD is a variable blocked data set (RECFM=VB) with a minimum recommended LRECL value [6] of 4096.

Input Data Set Specification

Allowable ddnames for the data sets that correspond to the input data sets are INDD1 through INDD255. The input data sets must be numbered consecutively. For example, if 25 input data sets are provided, they must be assigned ddnames INDD1 through INDD25. The utility processes the input data sets until a number is omitted. You must provide at least one input data set (INDD1).

The ddname number must correspond to the position of the input data set name in the data set name table (ICHRDSNT). That is, you must assign INDD1 to the first data set, INDD2 to the second, and so on.

6. The LRECL for the output data set must be at least as large as the largest record created by IRRDBU00. IBM recommends choosing a larger value so that if the size of the output records increases when new fields are added, you do not have to change your data set allocations. IBM recommends a value of 4096.

IRRDBU00 Example

In this example, the database unload utility processes a database that has been split into three parts. The job control language (JCL) statements that invoke the utility are:

```
//USER01   JOB  Job card...
//UNLOAD   EXEC PGM=IRRDBU00,PARM=NOLOCKINPUT
//SYSPRINT DD   SYSOUT=*
//INDD1    DD   DISP=SHR,DSN=SYS1.RACFDB.PART1.COPY
//INDD2    DD   DISP=SHR,DSN=SYS1.RACFDB.PART2.COPY
//INDD3    DD   DISP=SHR,DSN=SYS1.RACFDB.PART3.COPY
//OUTDD    DD   DISP=SHR,DSN=SYS1.RACFDB.FLATFILE
```

Note: You must specify a parameter in the PARM field on the EXEC statement of the step executing IRRDBU00.

Allowable Parameters

When you run the database unload utility, one of the following parameters must be specified: NOLOCKINPUT, LOCKINPUT, or UNLOCKINPUT. You can abbreviate the parameter to N, L, or U, respectively. For the *least* impact to system performance, use a copy of your RACF database as input and specify the NOLOCKINPUT parameter.

When your system is RACF is enabled for sysplex communication and RACF is running in read-only mode, the *only* parameter allowed for IRRDBU00 is NOLOCKINPUT.

Using the backup copy of the RACF database is allowed. Using an active copy of the RACF database can affect system performance and it is not recommended.

The NOLOCKINPUT Parameter

This value allows the unload to be performed and does not change the state of the input database. If the database is locked, it remains locked. If it is unlocked, it remains unlocked.

For the least impact to system performance, use a copy of your RACF database as input and specify the NOLOCKINPUT parameter.

Attention

If you use NOLOCKINPUT on the *active* database, your unloaded database might contain inconsistencies.

The LOCKINPUT Parameter

This value ensures that the RACF database used as input is not updated by other jobs while the utility is running.

Note: Statistics are updated.

If you are running against an active RACF database, LOCKINPUT is recommended.

Specifying LOCKINPUT means updates are no longer allowed to an input data set until the utility ends. If the RACF database is *locked* and users logging on attempt to change their user profiles, the logon may not be allowed, depending on the change. Users may be unable to:

- Change the password
- Specify a correct password after specifying an incorrect one

- Successfully complete the first logon of the day

For the logon to be allowed, you must apply APAR OY56802 to TSO/E 2.3, TSO/E 2.3.1, or TSO/E 2.4.

If you run IRRDBU00 and use LOCKINPUT, any activity that tries to update the RACF database (such as users logging on and changing passwords or batch jobs allocating new data sets requiring the creation of RACF profiles) fails with either an ABEND483 RC50 or ABEND485 RC50.

When using LOCKINPUT against an active database, do not schedule maintenance that runs past midnight. If RACF is not running in data sharing mode and the RACF database remains locked past midnight, new jobs cannot be submitted and users cannot log on unless you disable the gathering of logon statistics by issuing a SETROPTS NOINITSTATS command. All steps that require a locked database must be performed on the same calendar day. This is because RACF updates both primary and backup logon statistics each day.

The database unload utility *unlocks* the RACF database after processing with LOCKINPUT specified if the database was unlocked when the utility started. The unload utility output is for report generation and does not replace the input database, which is your primary, active, RACF database.

This is different from the IRRUT400 utility, which keeps the input database locked and creates a new output database. This is done to maintain integrity between the input database and the output database.

The UNLOCKINPUT Parameter

UNLOCKINPUT is used to unlock a database that had previously been locked by the LOCKINPUT parameter. This action enables your input database and allows it to be updated.

No data unloading is done when this parameter is used.

Using the Database Unload Utility Output Effectively

The output file from the database unload utility can be:

- Viewed directly
- Used as input to your own programs
- Manipulated with sort/merge utilities
- Used as input to a database management system. Installations can produce reports that are tailored to their requirements.

Sort/Merge Programs

The database unload utility processes all of the profiles in the input database. If you want a subset of the output records, you can use a standard utility such as DFSORT to select them. For example, the following DFSORT control statements select the Group Basic Data records (type 0100) and User Basic Data records (type 0200). All other record types are excluded.

```
SORT    FIELDS=(5,4,CH,A,10,20,CH,A)
INCLUDE COND=(5,4,CH,EQ,C'0100',OR,5,4,CH,EQ,C'0200')
OPTION  VLSHRT
```

Using Database Unload Utility Output with the DFSORT ICETOOL

IBM's DFSORT product provides a reporting facility called ICETOOL. You can create ICETOOL reports based on output files from the RACF database unload utility (IRRDBU00) or the SMF data unload utility (IRRADU00). The SYS1.SAMPLIB

member IRRICE contains DFSORT statements for record selection and ICETOOL statements for report formatting for a wide variety of reports. The IEBUPDTE utility processes the IRRICE member and creates a partitioned data set (PDS) that contains two PDS members for each report. The two members contain:

1. The report format
2. The record selection criteria

Attention: For information about the RACF database unload utility (IRRDBU00), see "Using the RACF Database Unload Utility (IRRDBU00)" on page 322. For information about the SMF data unload utility (IRRADU00), see *z/OS SecureWay Security Server RACF Auditor's Guide*.

The Report Format: The report format has a 1–4 character name (for example, UGRP). It contains the ICETOOL statements that control report format and record summary information, such as SORT, COPY, DISPLAY, and OCCURS statements. An example of a report format member is shown in Figure 16. This is the report format member UGRP, which is the report format for the "Users With Extraordinary Group Authorities" report.

```
//*******************************************************************
//* Name: UGRP                                                      *
//*                                                                 *
//* Find all of the user IDs which have extraordinary RACF privileges, *
//* such as SPECIAL, OPERATIONS, and AUDITOR at the group level.    *
//*******************************************************************
 SORT    FROM(DBUDATA)  TO(TEMP0001) USING(RACF)
 DISPLAY FROM(TEMP0001) LIST(PRINT) -
         PAGE -
         TITLE('UGRP: Users With Extraordinary Group Authorities') -
         DATE(YMD/) -
         TIME(12:) -
         BLANK -
         ON(10,8,CH)  HEADER('User ID') -
         ON(19,8,CH)  HEADER('Group ID') -
         ON(88,4,CH)  HEADER('Group Special') -
         ON(93,4,CH)  HEADER('Group Operations') -
         ON(113,4,CH) HEADER('Group Auditor')
```

Figure 16. Member UGRP: Users With Extraordinary Group Authorities report format statements

See *z/OS SecureWay Security Server RACF Macros and Interfaces* for the conversion rules of the database unload utility.

The Record Selection Criteria: The name of the member containing the record selection criteria is the report member name followed by CNTL (e.g. UGRPCNTL). Record selection is performed using DFSORT control statements, such as SORT and INCLUDE. The SORT command is used to select and sort records. The INCLUDE command is used to specify conditions required for records to appear in the report.

An example of a record selection member is shown in Figure 17 on page 328. This is the report selection member UGRPCNTL, which contains the selection criteria for the "Users With Extraordinary Group Authorities" report. In this example, we are including only User Connect Data records (record type 0205) when the user has the group-SPECIAL, group-OPERATIONS or group-AUDITOR attribute.

```
SORT    FIELDS=(10,08,CH,A)
INCLUDE COND=(5,4,CH,EQ,C'0205',AND,
             (88,3,CH,EQ,C'YES',OR,
              93,3,CH,EQ,C'YES',OR,
              113,3,CH,EQ,C'YES'))
OPTION  VLSHRT
```

Figure 17. Member UGRPCNTL: Users With Extraordinary Group Authorities Report record selection statements

See *z/OS SecureWay Security Server RACF Macros and Interfaces* for record format information for the output records of the IRRADU00 and IRRDBU00 utilities. See *DFSORT Application Programming Guide R14* for the complete details of the DFSORT statements.

An Important Note on Column Numbers

Both IRRADU00 and IRRDBU00 create records that are variable-length. Variable-length records have a four-byte record descriptor word (RDW) describing the length of the record. DFSORT considers the RDW to be part of the selectable record columns. This means that you must add 4 to any of the field positions identified for the IRRADU00 and IRRDBU00 records described in *z/OS SecureWay Security Server RACF Macros and Interfaces*. In the example in Figure 17, the IRRDBU00 field for record type 0205 is defined in *z/OS SecureWay Security Server RACF Macros and Interfaces* as beginning at record position 1. We add 4 to this position to get 5, the value that we must use in both the DFSORT INCLUDE statement for record selection and the ICETOOL ON operand to select the fields for the report.

Using the RACFICE Procedure to Generate Reports: You can invoke the ICETOOL utility with the RACFICE procedure contained in the IRRICE member of SYS1.SAMPLIB. It simplifies the JCL required to execute ICETOOL reports and contains JCL symbolic variables that represent the input to the RACFICE procedure. These variables are:

DBUDATA Output of IRRDBU00 that is being used as input

ADUDATA Output of IRRADU00 that is being used as input

REPORT The name of the report that is being generated

See “Reports Based on the Database Unload Utility (IRRDBU00)” on page 329 for a list of the reports based on IRRDBU00 output that are shipped with this support. See *z/OS SecureWay Security Server RACF Auditor’s Guide* for a list of the reports based on IRRADU00 output that are shipped with this support.

See “Creating Customized Reports” on page 331 for information about creating your own reports.

You don’t need to specify each of these variables every time you execute the RACFICE procedure. For example, if you specify the default IRRDBU00 and IRRADU00 data sets in the RACFICE procedure, you create a report (shown in Figure 18 on page 329) that lists all user IDs with extraordinary RACF group authorities with the following JCL:

```
//jobname  JOB  Job card...
//stepname EXEC RACFICE,REPORT=UGRP
```

If the default IRRDBU00 or IRRADU00 data sets are not correct, you can override them. For example, if the IRRDBU00 output is in the data set USER01.TEST.IRRDBU00 and the IRRADU00 output is in the data set USER01.TEST.IRRADU00, you should enter:

```
//jobname  JOB  Job card...
//         SET  ADUDATA=USER01.TEST.IRRADU00
//         SET  DBUDATA=USER01.TEST.IRRDBU00
//stepname EXEC RACFICE,REPORT=UGRP
```

```
1- 1 -   UGRP: Users With Extraordinary Group Authorities    99/04/13

 User ID    Group ID    Group Special  Group Operations  Group Auditor
 --------   --------    -------------  ----------------  -------------
 LCLAUDIT   GROUP1      NO             NO                 YES
 LCLOPER    GROUP1      NO             YES                NO
 LCLSPEC    GROUP1      YES            NO                 NO
 UCAUDR$Y   GPCONNEC    NO             NO                 YES
 UCOPER$Y   GPCONNEC    NO             YES                NO
 UCSPEC$Y   GPCONNEC    YES            NO                 NO
```

Figure 18. Report of All Users with Extraordinary Group Authorities

Reports Based on the Database Unload Utility (IRRDBU00): The following reports are based on the output of IRRDBU00. You can find a sample of each report in SYS1.SAMPLIB.

Name	Description
ALDS	Data set profiles which have IDs on the standard access list with ALTER authority. **Value:** Identifies users who can alter the access list of the profile.
ASOC	Users who have explicit RACF Remote Sharing Facility (RRSF) associations defined. **Value:** Identifies users who can direct commands.
BGGR	Discrete general resource profiles with generic characters. **Value:** Finds profiles which aren't protecting what you think that they are protecting.
CCON	Count of user's connections, flagging those users with more than "x" connections. **Value:** Helps find a performance bottleneck caused by excessive group connections.
CGEN	Count of general resource profiles. **Value:** Identifies basic characteristics of the RACF database.
CPRO	Count of user, group, and data set profiles. **Value:** Identifies basic characteristics of the RACF database.
CONN	User IDs with group authorities above USE. **Value:** Identifies users with additional privileges.
CUG$	Group names of all universal groups, listing their owners and creation dates. **Value:** Identifies universal groups that may have members who do not appear in the group's member list.
GRPM	User IDs of all members of a group, including a universal group, listing the owner of each connection, and group-related user

	attributes for each member. **Value:** Provides a complete member listing for universal groups, which is not available using the LISTGRP command.
IDSC	Dataset conditional access list entries with an ID(*) entry of other than NONE. **Value:** Identifies dataset profiles that allow any RACF-authenticated user to access data.
IDSS	Dataset standard access list entries with an ID(*) entry of other than NONE. **Value:** Identifies dataset profiles that allow any RACF-authenticated user to access data.
IGRC	General resource conditional access list entries with an ID(*) entry of other than NONE. **Value:** Identifies general resource profiles that allow any RACF-authenticated user to access data.
IGRS	General resource standard access list entries with an ID(*) entry of other than NONE. **Value:** Identifies general resource profiles that allow any RACF-authenticated user to access data.
OMVS	User IDs which have an OMVS segment defined. **Value:** Identifies users who can use z/OS UNIX with a non-default UID.
SUPU	z/OS UNIX “superusers” (UID of zero). **Value:** Identifies users who have extraordinary privileges within the z/OS UNIX environment.
UADS	Dataset profiles with UACCs other than NONE. **Value:** Identifies dataset profiles that allow any user to access data.
UAGR	General resource profiles, excluding profiles in the DIGTCERT class, with UACCs other than NONE. **Value:** Identifies general resource profiles that allow any user to access data.
UGLB	User IDs with extraordinary global authorities. **Value:** Identifies users with extraordinary RACF authority.
UGRP	User IDs with extraordinary RACF group authorities. **Value:** Identifies users with extraordinary RACF authority.
UIDS	z/OS UNIX UIDs which are used more than once. **Value:** Identifies z/OS UNIX users who are sharing authority characteristics.
URVK	User IDs which are currently revoked. **Value:** Identifies users who have had a revocation performed.

In addition, the following reports demonstrate advanced ICETOOL techniques:

Name	**Description**
$CFQG	A count of the number of fully-qualified generic profiles that are defined for each high-level qualifier (HLQ). **Value:** Identifies users who have defined an excessive number of fully qualified generic profiles.
$CHLQ	A count of the number of generic profiles that are defined for each high-level qualifier (HLQ). **Value:** Identifies a potential performance bottleneck.
$ULAST90	Identifies the user profiles which have been created within the past 90 days. **Value:** Shows recent administrative activity.

Note that these reports ($CFQC, $CHLQ, and $ULAST90) are standalone reports and are not run using the RACFICE PROC.

Creating Customized Reports: You can create your own reports using the RACFICE procedure by following these steps:

1. Identify the records that you want for the report.
   a. Define the DFSORT statements for the record selection criteria.
   b. Place them in the RACFICE data set with a unique member name consisting of a 1–4 character report identifier followed by CNTL.

   If there is an existing report that has similar selection criteria, use it as a model. For example, if you want to report all the access records created when users PATTY, MAXINE, and LAVERNE accessed resources, you need to create DFSORT selection statements that look like Figure 19 and store them in your RACFICE report data set as the PMLCNTL record selection criteria.

```
INCLUDE COND=(63,8,CH,EQ,C'PATTY',OR,
       63,8,CH,EQ,C'MAXINE',OR,
       63,8,CH,EQ,C'LAVERNE')
OPTION  VLSHRT
```

Figure 19. Customized Record Selection Criteria

   Note the similarity of this record selection criteria to the “Users With Extraordinary Group Authorities Report” record selection criteria shown in Figure 17 on page 328.

   See *DFSORT Application Programming Guide R14* for the complete details of the DFSORT statements.
2. Identify the report format you want to use.
   a. Define the ICETOOL statements for the report format.
   b. Place them in the RACFICE data set with a 1–4 character report identifier that you chose.

   If there is an existing report that has similar report format, use it as a model. For example, if you wanted your report to contain the user ID, job name, date, time, and status of the access you could use the ICETOOL report statements shown in Figure 20, and store them in your RACFICE report data set as the PML report format.

```
COPY    FROM(ADUDATA)  TO(TEMP0001) USING(RACF)
DISPLAY FROM(TEMP0001) LIST(PRINT) -
        PAGE -
        TITLE('Patty, Maxine, and Laverne's Accesses') -
        DATE(YMD/) -
        TIME(12:) -
        BLANK -
        ON(63,8,CH)  HEADER('User ID') -
        ON(5,8,CH)   HEADER('Event') -
        ON(12,8,CH)  HEADER('Qualifier') -
        ON(23,8,CH)  HEADER('Time') -
        ON(32,10,CH) HEADER('Date') -
        ON(184,8,CH) HEADER('Job Name')
```

Figure 20. Customized Report Format

   Note the similarity of this report format to the “Users With Extraordinary Group Authorities” report format shown in Figure 16 on page 327.

For complete details on the ICETOOL statements, see *DFSORT Application Programming Guide R14*.

3. Update the report JCL to invoke the RACFICE procedure with the 1–4 character report identifier you chose, as shown in Figure 21.

```
//jobname  JOB  Job card...
//stepname EXEC RACFICE,REPORT=PML
```

Figure 21. Customized Report JCL

Relational Databases

Much of the function of the database unload utility is not realized until the data it creates is loaded into a relational database management system (DBMS) such as DB2.

Using the Database Unload Utility Output with DB2

You can use the DB2 load utility or its equivalent to process the records produced by the database unload utility. The definition and control statements for a DB2 utilization of the output, all of which are contained in SYS1.SAMPLIB, are as follows:

- Sample data definition language (DDL) statements to define the relational presentation of the RACF database and sample DB2 definitions which perform database and index creation. These are contained in member RACDBUTB.
- Sample control statements for the DB2 load utility that map the output from the database unload utility (IRRDBU00). These are contained in member RACDBULD.
- Sample structured query language (SQL) queries that perform the following queries. These are contained in member RACDBUQR.
  - Listing all of the members of a group, including a universal group
  - Listing all of the users with the SPECIAL attribute
  - Finding all of the groups to which a user is connected
  - Finding all of the data set access lists that contain user IDs that are no longer valid
  - Listing of z/OS UNIX user identifiers (UIDs) with associated user ID and programmer name
  - Listing of z/OS UNIX group identifiers (GIDs) with associated group name
  - Listing of UIDs including only those users who are connected to a group that has a GID (for each UID, the user ID and programmer name are listed)

For more information on DB2, see:
- *DB2 Administration Guide*
- *DB2 SQL Reference*

Steps for Using IRRDBU00 Output with DB2: To create and manage a DB2 database that contains the output from the database unload utility, you must:
1. Create one or more DB2 databases.
2. Create one or more DB2 table spaces.
3. Create DB2 tables.
4. Create the DB2 indexes.
5. Load data into the tables.
6. Reorganize the data in the tables (optional).
7. Create performance statistics (optional).
8. Delete table data (optional).

The first three steps are initial setup, and you can choose to run them once. When you get new data to import into the DB2 database, you delete your current table data. You then reload and reorganize your tables and create the performance statistics.

The following sections show examples of the DB2 utility input for these functions.

Creating a DB2 Database for Unloaded RACF Data: A DB2 database names a collection of table spaces. The following SQL statement creates a DB2 database for the output of the database unload utility:

```
CREATE DATABASE databasename
```

where *databasename* is supplied by the user.

Creating a DB2 Table Space: A table space is one or more data sets in which one or more tables are stored. Figure 22 contains examples of SQL statements that create a table space. There are other methods of allocating a table space. For details, see the DB2 documentation.

Member RACDBUTB in SYS1.SAMPLIB contains statements that create a table space.

```
CREATE TABLESPACE tablespacename IN databasename
       LOCKSIZE TABLESPACE
       SEGSIZE  64
       PCTFREE  0
       USING STOGROUP storagegroup
          PRIQTY 2000
          SECQTY 500
          CLOSE  NO
          ;
```

Figure 22. Sample SQL Utility Statements: Defining a Table Space

The user must supply the name of the table space (*tablespacename*) and the storage group (*storagegroup*). The sample shows a value of 64 for SEGSIZE, 2000 for PRIQTY, and 500 for SECQTY.

The samples in RACDBUTB put all of the tables into one table space. The sample also suggests using a segment size because segmented table spaces improve performance. Users may want to define their own table spaces rather than use table spaces that are defined by the storage group.

Installations have a number of other options, such as the number of table spaces to use, the type of spaces, and the security for the data. They may want to keep the number of tables per table space fairly small for better performance and may want to consider putting the larger tables into separate table spaces.

Creating the DB2 Tables: After the database and the table space are created, SQL statements that define the tables are executed. Figure 23 on page 335 contains an example of the SQL statements that are required to create a table for the Group Basic Data record of the database unload utility.

Member RACDBUTB in SYS1.SAMPLIB contains examples that create separate tables for each record type that is produced by the database unload utility. The user must supply the user ID (*userid*).

```
CREATE TABLE userid.GROUP_BD (
     GPBD_NAME          CHAR(8)   NOT NULL,
     GPBD_SUPGRP_ID     CHAR(8),
     GPBD_CREATE_DATE   DATE,
     GPBD_OWNER_ID      CHAR(8)   NOT NULL,
     GPBD_UACC          CHAR(8)   NOT NULL,
     GPBD_NOTERMUACC    CHAR(1)   NOT NULL,
     GPBD_INSTALL_DATA  CHAR(254),
     GPBD_MODEL         CHAR(44)
     )
     IN databasename.tablespacename
     ;
```

Figure 23. Sample SQL Utility Statements: Creating a Table

Creating the DB2 Indexes: DB2 performance improves with the use of indexes. Member RACDBUTB in SYS1.SAMPLIB creates an index for every primary key and every foreign key identified in the record types. Figure 24 on page 336 contains sample statements to create the indexes for the Group Basic Data record.

Notes:

1. Users on DB2 Version 2 Release 2 may wish to code CLOSE NO if the tables are accessed often.
2. Users on DB2 Version 2 Release 3 may wish to code CLOSE YES.

```
CREATE UNIQUE INDEX userid.GROUP_BD_IX1
       ON userid.GROUP_BD
          (GPBD_NAME)
          USING   STOGROUP storagegroup
          PRIQTY  100
          SECQTY  50
          CLUSTER
          PCTFREE 0
          CLOSE   NO
          ;
CREATE UNIQUE INDEX userid.GROUP_BD_IX2
       ON userid.GROUP_BD
          (GPBD_NAME, GPBD_SUPGRP_ID)
          USING   STOGROUP storagegroup
          PRIQTY  100
          SECQTY  50
          PCTFREE 0
          CLOSE   NO
          ;
CREATE INDEX userid.GROUP_BD_IX3
       ON userid.GROUP_BD
          (GPBD_OWNER_ID)
          USING   STOGROUP storagegroup
          PRIQTY  100
          SECQTY  50
          PCTFREE 0
          CLOSE   NO
          ;
CREATE INDEX userid.GROUP_BD_IX4
       ON userid.GROUP_BD
          (GPBD_MODEL)
          USING   STOGROUP storagegroup
          PRIQTY  100
          SECQTY  50
          PCTFREE 0
          CLOSE   NO
          ;
```

Figure 24. Sample SQL Utility Statements: Creating Indexes

Loading the DB2 Tables: Figure 25 shows the statements that are required to load the Group Basic Data record. The RACDBULD member of SYS1.SAMPLIB contains statements that load all of the record types produced by the database unload utility.

```
LOAD DATA
     INDDN  tablespacename
     RESUME YES
     LOG    NO
     INTO   TABLE userid.GROUP_BD
     WHEN(1:4)='0100' (
         GPBD_NAME            POSITION(006:013)  CHAR(8),
         GPBD_SUPGRP_ID       POSITION(015:022)  CHAR(8)
         GPBD_CREATE_DATE     POSITION(024:033)  DATE EXTERNAL(10),
         GPBD_OWNER_ID        POSITION(035:042)  CHAR(8),
         GPBD_UACC            POSITION(044:051)  CHAR(8),
         GPBD_NOTERMUACC      POSITION(053:053)  CHAR(1),
         GPBD_INSTALL_DATA    POSITION(058:311)  CHAR(254),
         GPBD_MODEL           POSITION(314:357)  CHAR(44)
         )
```

Figure 25. DB2 Utility Statements Required to Load the Tables

Note: You can choose not to load some of the tables.

Reorganizing the Unloaded RACF Data in the DB2 Database: Queries are processed faster if they are performed against an organized database. The DB2 utility statement required to reorganize the database is:

```
REORG TABLESPACE databasename.tablespacename
```

Creating Optimization Statistics for the DB2 Database: Queries are processed faster if they are performed against an organized database for which DB2 has collected performance statistics. The DB2 utility statement required to create these statistics is:

```
RUNSTATS TABLESPACE databasename.tablespacename
```

Deleting Data from the DB2 Database: Before you reload the database with new data, you should delete the old data. This can be done in several ways:

1. Use the DROP TABLE statement for each table you want to delete.
2. Use the DROP TABLESPACE statement for each tablespace.
3. Delete all of the records in each table.

   Figure 26 shows the sample SQL statements that delete the group record data from the tables.

```
DELETE FROM userid.GROUP_BD           ;
DELETE FROM userid.GROUP_DFP_DATA     ;
DELETE FROM userid.GROUP_INSTALL_DATA ;
DELETE FROM userid.GROUP_SUBGROUPS    ;
DELETE FROM userid.GROUP_MEMBERS      ;
```

Figure 26. DB2 Utility Statements Required to Delete the Group Records

DB2 Table Names: Member RACDBULD in SYS1.SAMPLIB creates DB2 tables for each record type. Table 26 provides a useful reference of record type, record name, and DB2 table name.

For more information, see “Database Unload Utility Output Samples” on page 340.

Table 26. Correlation of Record Type, Record Name, and DB2 Table Name

Record Type	Record Name	DB2 Table Name
0100	Group Basic Data	GROUP_BD
0101	Group Subgroups	GROUP_SUBGROUPS
0102	Group Members	GROUP_MEMBERS
0103	Group Installation Data	GROUP_INSTALL_DATA
0110	Group DFP Data	GROUP_DFP_DATA
0120	Group OMVS Data	GROUP_OMVS_DATA
0130	Group OVM Data	GROUP_OVM_DATA
0140	Reserved	
0141	Group TME Role	GROUP_TME_ROLE
0200	User Basic Data	USER_BD
0201	User Categories	USER_CATEGORIES
0202	User Classes	USER_CLASSES
0203	User Group Connections	USER_GROUPS
0204	User Installation Data	USER_INSTALL_DATA
0205	User Connect Data	USER_CONNECT_DATA
0206	User RRSF Data	USER_RRSF_DATA
0207	User Certificate Data	USER_CERT_DATA
0208	User Associated Mappings	USER_NMAP_DATA
0210	User DFP Data	USER_DFP_DATA

Table 26. Correlation of Record Type, Record Name, and DB2 Table Name (continued)

Record Type	Record Name	DB2 Table Name
0220	User TSO Data	USER_TSO_DATA
0230	User CICS Data	USER_CICS_DATA
0231	User CICS Operation Classes	USER_CICS_OPCLASS
0240	User LANGUAGE Data	USER_LANGUAGE_DATA
0250	User OPERPARM Data	USER_OPERPARM_DATA
0251	User OPERPARM Scope	USER_OPERPARM_SCOP
0260	User WORKATTR Data	USER_WORKATTR_DATA
0270	User OMVS Data	USER_OMVS_DATA
0280	User NETVIEW Data	USER_NETV_DATA
0281	User NETVIEW Operation Classes	USER_NETV_OPCLASS
0282	User NETVIEW Domains	USER_NETV_DOMAINS
0290	User DCE Data	USER_DCE_DATA
02B0	User LNOTES Data	USER_LNOTES_DATA
02C0	User NDS Data	USER_NDS_DATA
02D0	User KERB Data	USER_KERB_DATA
0400	Data Set Basic Data	DS_BD
0401	Data Set Categories	DS_CATEGORIES
0402	Data Set Conditional Access	DS_COND_ACCESS
0403	Data Set Volumes	DS_VOLUMES
0404	Data Set Access	DS_ACCESS
0405	Data Set Installation Data	DS_INSTALL_DATA
0410	Data Set DFP Data	DS_DFP_DATA
0420	Reserved	
0421	Data Set TME Role	DS_TME_ROLE
0500	General Resource Basic Data	GENR_BD
0501	General Resource Tape Volumes	GENR_TAPE_VOLUMES
0502	General Resource Categories	GENR_CATEGORIES
0503	General Resource Members	GENR_MEMBERS
0504	General Resource Volumes	GENR_VOLUMES
0505	General Resource Access	GENR_ACCESS
0506	General Resource Installation Data	GENR_INSTALL_DATA
0507	General Resource Conditional Access	GENR_COND_ACCESS
0508	General Resource Filter Data	GENR_FILTER_DATA
0510	General Resource Session Data	GENR_SESSION_DATA
0511	General Resource Session Entities	GENR_SESSION_ENT
0520	General Resource DLF Data	GENR_DLF_DATA
0521	General Resource DLF Job Names	GENR_DLF_JOBNAMES
0540	General Resource STDATA Data	GENR_STDATA_DATA
0550	General Resource SVFMR Data	GENR_SVFMR_DATA
0560	General Resource Certificate Data	GENR_GRCERT_DATA
0561	General Resource Certificate References Data	GENR_CERTR_DATA
0562	General Resource Key Ring Data	GENR_KEYR_DATA
0570	General Resource TME Data	GENR_TME_DATA
0571	General Resource TME Children	GENR_TME_CHILDREN
0572	General Resource TME Resource	GENR_TME_RESOURCE
0573	General Resource TME Group	GENR_TME_GROUP
0574	General Resource TME Role	GENR_TME_ROLE
0580	General Resource KERB Data	GENR_KERB_DATA

Processing User Revocation Information

The RACF database unload utility (IRRDBU00) creates as output a representation of the data in the RACF database. Only a minimal amount of interpretation is done. This differs from some RACF commands such as LISTUSER and LISTGRP, which interpret the data they find in the RACF database. You may notice this difference among the output of IRRDBU00, the output of the LISTUSER command, and the output of the LISTGRP command when reviewing the information that describes revocation status.

For users and their connections to groups, RACF stores three pieces of information describing the revocation status:

revoke_date The date from which the user is revoked

resume_date The date on which the user is no longer revoked

revoked_flag A flag indicating if the user has been revoked

At logon or job initiation, RACF compares the current date with the *revoke_date* and *resume_date.* If the current date falls between them, the logon or job initiation is not allowed or the connection with the group is not considered valid.

LISTUSER and LISTGRP perform similar checks. For example, if the date on which the LISTUSER command is issued falls between the *revoke_date* and the *resume_date*, LISTUSER reports that the user is revoked. If the date on which the LISTUSER command is issued does not fall between the *revoke_date* and the *resume_date*, LISTUSER indicates that the user is not revoked even if the *revoke_date*, *resume_date*, and *revoked_flag* are set in the RACF database.

Note: It is possible to have no data for the *revoke_date* and *resume date*.

Because IRRDBU00 does not have a reference date such as the current date, it cannot interpret the *revoke_date*, *resume_date*, and *revoked_flag* information with a reference date. IRRDBU00 unloads the values as they are specified in the RACF database. This means that if you write a query that just checks the *revoked_flag*, the results differ from LISTUSER and LISTGRP.

You can incorporate a date check into your queries that performs the same checks as the LISTUSER and LISTGRP commands. Figure 27 on page 340 shows a structured query language (SQL) fragment to do this test for the user-revoked status. Note that `CURRENT DATE` can be replaced with any valid DB2 date value.

```
SELECT
  ....
  WHERE
     (CURRENT DATE >= USBD_REVOKE_DATE   AND
      CURRENT DATE < USBD_RESUME_DATE)
    OR
     (CURRENT DATE < USBD_RESUME_DATE     AND
      USBD_REVOKE_DATE IS NULL)
    OR
     (CURRENT DATE >= USBD_REVOKE_DATE   AND
      USBD_RESUME_DATE IS NULL)
    OR
     (USBD_REVOKE_DATE IS NULL AND
      USBD_RESUME_DATE IS NULL AND
      USBD_REVOKE='Y')
      ....
```

Figure 27. An SQL Fragment to Process Revoke and Resume Dates

Database Unload Utility Output Samples

A relational database management system such as DB2 can be used with the Query Management Facility (QMF) to create reports.

A report many installations find useful is a list of all of the data set profiles that contain an non-valid ID in the access list. This situation occurs when a security administrator deletes a user ID or group name without deleting the authorities the ID may have had in the RACF database.

To search the data set access list for user IDs and group names that do not have user or group profiles in the database, perform the following steps:

1. Create a query that compares the entries in the access list with a list of valid user IDs or group names. A sample SQL query is provided in Figure 28 on page 341.
2. Format the results of the query as provided by the QMF form in Figure 29 on page 341.

The resulting report is shown in Figure 30 on page 342.

Note: If you use the IRRUT100 utility to check the references to a user or group name, IRRUT100 requires that the user or group name be known. The sample query shown here does not have such a requirement. It finds *all* user IDs or group names that are not valid.

When your RACF database is unloaded, the IRRDBU00 utility creates a Data Set Access Record (record type 404) for each user ID or group name in the access list of each data set.

When you load your IRRDBU00 output into DB2, an AUTH_IDS table is created that contains the name of every valid user ID and group name.

SQL Query: The sample SQL query compares the ID in the data set access record (`DSACC_AUTH_ID`) with the list of valid user and group names (in AUTH_IDS). When a user ID is found that is not a valid user ID or group name, it is listed. The query also lists the data set profile name, the authority that the user has, and the access count.

```
-------------------------------------------------------------------------
-- Description: Check all of the data set standard access lists and   --
--              verify that each user ID is a valid user or           --
--              group name                                            --
--                                                                    --
-- Tables Accessed: SQL                                               --
--                  "DS_ACCESS"   - A list of data set authorities    --
--                  "AUTH_IDS"   - A list of valid users/groups       --
--                                                                    --
-------------------------------------------------------------------------
     SELECT
            DSACC_NAME
           ,DSACC_AUTH_ID
           ,DSACC_ACCESS
           ,DSACC_ACCESS_CNT
     FROM
           USER01.DS_ACCESS X
     WHERE NOT EXISTS
           ( SELECT *
              FROM
                     USER01.AUTH_IDS
              WHERE
                     X.DSACC_AUTH_ID=AUTHID_NAME
                      )
     AND
           X.DSACC_AUTH_ID¬='*'
     ORDER BY 1
     ;
```

Figure 28. A Sample SQL Query

QMF Form: If the SQL query shown in Figure 28 is processed using QMF, the data that is returned can be processed into a report. Figure 29 shows a report or forms definition. It creates the report shown in Figure 30 on page 342 entitled “Data Set Profiles With Access Lists Containing User IDs or Group Names That Are Not Valid”.

```
COLUMNS:               Total Width of Report Columns: 80

NUM COLUMN HEADING                          USAGE  INDENT WIDTH EDIT SEQ
--- ------------------------------------ ------ ------ ----- ---- ---
 1 DSACC_NAME                            BREAK1 2      44    C    1
 2 DSACC_AUTH_ID                               2      8     C    2
 3 DSACC_ACCESS                                2      9     C    3
 4 DSACC_ACCESS_CNT                            2      11    L    4


PAGE:    HEADING  ===> DATA SET PROFILES WITH ACCESS LISTS CONTAINING
                       USER IDS OR GROUP NAMES THAT ARE NOT VALID
       FOOTING  ===>
FINAL:   TEXT     ===>
BREAK1:  NEW PAGE FOR BREAK? ===> NO
         FOOTING  ===>
BREAK2:  NEW PAGE FOR BREAK? ===> NO
         FOOTING  ===>
OPTIONS: OUTLINE? ===> YES          DEFAULT BREAK TEXT? ===> NO
```

Figure 29. A Sample QMF Form

Report Output: Figure 30 shows the report that results from the SQL query shown in Figure 28 on page 341 and the QMF form shown in Figure 29 on page 341. Not all of the resulting rows are shown.

```
DATA SET PROFILES WITH ACCESS LISTS CONTAINING USER IDS OR GROUP
NAMES THAT ARE NOT VALID

DSACC_NAME                      DSACC_AUT DSACC_ACC DSACC_ACCES
------------------------------- --------  --------- -----------
TERRYS.WORK.CNTL                WILLIEC   READ                3
                                JEFFK     READ                2
                                JOHNL     READ                2
                                DIANEB    READ                4
                                SLICK     READ                5
                                DICKH     READ                2
                                CHUCKS    READ                1
                                ERICAK    READ                2
                                LIZS      READ                3
                                TONYL     READ                3
                                JOES      READ                6

GENEK.DOC.TEXT                  SUSANK    READ                4
                                JIMV      READ                2

ELVIS.TEST.DATA                 JOEC      READ               10

ACKAWAY.DESIGN.DATA             SUET      READ                3

HILDE.*                         STINGS    UPDATE              6
                                TIMS      READ                1



04/31/2000 04:26 PM                                    PAGE 10
```

Figure 30. A Sample Report

Using the RACF Remove ID Utility (IRRRID00)

The RACF remove ID utility (IRRRID00) can help you keep your RACF database current. You can use this utility to remove all references to group IDs and user IDs that no longer exist in or are about to be removed from the RACF database. Also, you can specify a replacement ID for those IDs that will be removed.

The remove ID utility processes the output of the RACF database unload utility (IRRDBU00). You need to have read access to this output. To get this output, run the database unload utility against a copy of the RACF database. See for more information.

The remove ID utility:

- Uses the DFSORT utility (or an equivalent program) to create lists of IDs from the output of the database unload utility or from user input in a SYSIN file.
- Compares these IDs to the user IDs and group names contained in such RACF data fields as:
  - Standard access list
  - Conditional access list
  - Profile names in certain general resource member classes
  - OWNER fields
  - NOTIFY fields
  - APPLDATA fields of certain general resource profiles

Note: See “Finding Residual IDs” on page 347 for more information about the fields searched by the remove ID utility.

- Generates output consisting of commands that change or remove the references to residual authority IDs or to the IDs you specify. For example, the output could include:
  - PERMIT commands to delete references on an access list
  - Commands to delete profiles, for profile names that contain the ID value
  - Commands to change references to other values, for references requiring that an ID value be specified

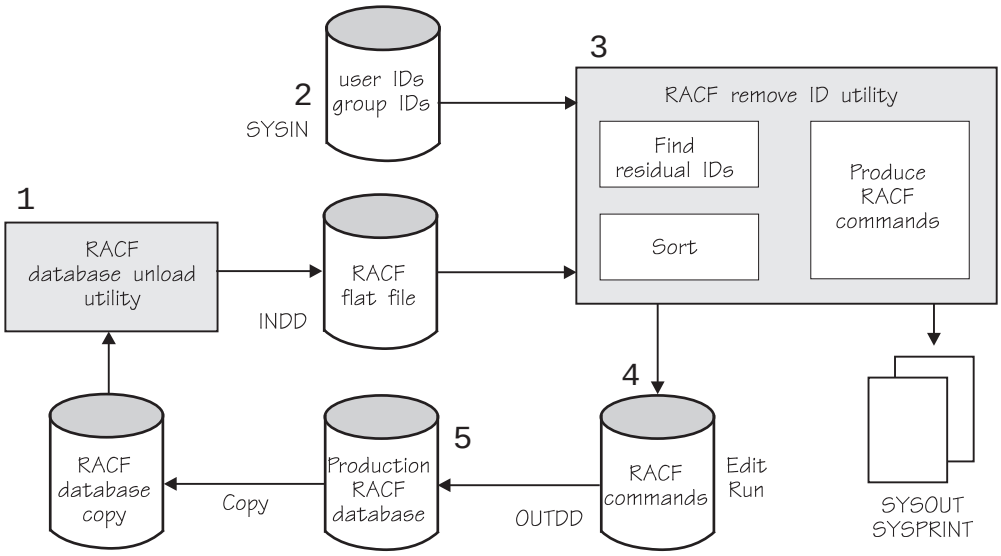


Figure 31. Using the Remove ID Utility

As shown in Figure 31, here’s how to use the remove ID utility:

1. Use the database unload utility to produce a *flat file.* This file is the main input to the remove ID utility. You should use a *copy* of the production RACF database as input to the database unload utility.
2. Optionally, you can specify a SYSIN file.

   If this file is empty or does not contain any valid input, or if it is allocated as DUMMY, the remove ID utility searches for residual references to user IDs or group names that do not exist as a user profile or a group profile. See “Running IRRRID00 with an Empty SYSIN” on page 348 for an example.
3. The remove ID utility does one of the following:
   a. Finds the residual IDs, sorts them, and then uses this list of IDs to produce output that contains the appropriate RACF commands.

      See “Finding Residual IDs” on page 347 for more information about this step.
   b. Uses a list of user IDs and group names that are specified in the SYSIN file to produce output that contains the appropriate RACF commands.

      See “Creating Commands to Remove IDs” on page 348 for more information about this step.
4. The remove ID utility creates an OUTDD file, which contains commands to change or remove the occurrences of these IDs.

You should review the commands the remove ID utility generates and, if necessary, edit them.

If you run the remove ID utility with no SYSIN file, or do not specify a replacement ID, the output shows any references to an ID that requires a replacement as **?***id*. This might be the case, for example, in places where a residual user ID was the owner of other profiles. You should change all occurrences of **?***id*, if any, to an *existing* user ID or group name.

5. As long as you have sufficient authority, you can now run these commands on the production RACF database. See *z/OS SecureWay Security Server RACF Command Language Reference* for the specific authority requirements for RACF commands.

Notes:

1. The remove ID utility deals with profiles in the RACF database. So, keep in mind that the remove ID utility does not produce any commands to delete, or rename the resources these profiles protect. You must delete, rename, or make sure other profiles protect those resources that were once protected.

   You can use DFSMSdss to rename data sets for IDs that you will be removing from the RACF database.
2. If you delete profiles before you delete or rename the data sets themselves and PROTECTALL is in effect, you might need some extra authority to remove these data sets.
3. If you remove a user ID that had been cross-linked with a DCE principal, contact the cell's DCE administrator to determine whether the DCE principal should be deleted from the cell.
4. If a residual ID is found in a NOTELINK or NDSLINK profile, an RDELETE command will be produced to delete the profile. However, if the profile name contains lower case characters, the RDELETE command cannot be executed successfully. To delete the profile, you must issue an ADDUSER command for the user ID specifying the corresponding LNOTES SNAME or NDS UNAME. Then, a DELUSER can be issued to delete the user profile and the NOTELINK or NDSLINK profile.
5. If a user ID that you have specified in the SYSIN file is found in the name of a user profile containing an LNOTES, NDS, or DCE segment, IRRRID00 will produce a DELUSER command to delete the user profile, but it will not produce RDELETE commands to delete the corresponding NOTELINK, NDSLINK, or DCEUUIDS general resource profiles. Deletion of the user ID through DELUSER processing will cause the deletion of the corresponding general resource profiles.

IRRRID00 Job Control Statements

The following job control statements are needed to run the remove ID utility:

JOB Initiates the job.

It is recommended that you run IRRRID00 with a region size of 25M:

```
EXEC PGM=IRRRID00,REGION=25M
```

Note: The storage required to run IRRRID00 successfully depends on the size of the RACF database and other factors that are controlled by the sort utility your installation uses. If the job does not run because there is not enough storage available, try increasing the region size. If your installation has a large RACF database, a region size of 0M may be required:

```
EXEC PGM=IRRRID00,REGION=0M
```

EXEC Specifies the program name (PGM=IRRRID00) or the procedure name if the job control statements are in a procedure library.

SYSPRINT DD
Defines a sequential message data set for the messages produced by IRRRID00.

SYSOUT DD Defines a sequential message data set for the messages produced by the DFSORT utility or its equivalent.

SORTOUT DD
Defines a work data set that contains final list records. This data set should be approximately the same size as the data set allocated to INDD.

SYSUT1 DD Defines a work data set that contains intermediary records. This data set should be approximately the same size as the data set allocated to INDD.

INDD DD Defines the sequential input data set that contains the IRRDBU00 output being processed. This statement should refer to the same data set as the OUTDD statement does in the IRRDBU00 job.

OUTDD DD Defines the single sequential output data set. The output of IRRRID00 is a set of variable length records that contain the commands needed to delete or alter the references to the IDs. This data set must be allocated as a variable length data set, with a logical record length (LRECL) of at least 259. If a shorter LRECL is supplied, IRRRID00 changes the LRECL to 259.

When IRRRID00 opens the OUTDD data set, it verifies that the block size of the data set is at least 4 greater than the LRECL.

SYSIN DD Defines the sequential input data set that contains the list of user IDs or group names to search for. Each ID must be on its own record. The ID can be up to 8 characters in length. It will be truncated if longer than 8 characters.

Optionally, you can specify a replacement ID. The replacement ID must be on the same input record, separated from the original ID by at least 1 blank. The replacement ID can be up to 8 characters in length. It will be truncated if longer than 8 characters.

IRRRID00 accepts both fixed length records (RECFM=F or RECFM=FB) and variable length records (RECFM=V or RECFM=VB) either with or without record numbers. To allow for record numbers, IRRRID00 always ignores columns 1–8 if the SYSIN records are variable length, and ignores the last eight columns if the records are fixed length. In addition, IRRRID00 ignores blank records.

Notes:

1. The SYSIN DD is optional.

   If SYSIN is specified, IRRRID00 does not validate the ID being searched for or the replacement ID.

   If it is not specified or if it points to a data set that does not contain a list of user IDs or group names, a message is issued to SYSPRINT and a search is performed for all references to IDs that no longer exist.

2. The percent (**%**) and asterisk (*****) are processed as regular characters; no generic processing will be performed. A period (**.**) should not be used but will be accepted; a match of IDs will not occur in most cases.
3. Some of the DD names shown are for DFSORT only. If you are using an equivalent product, refer to that product's documentation for the DD names to use.

Searching for All Residual References

To search for all references to IDs that no longer exist, run IRRRID00 with no IDs specified. You can do this either by not allocating the SYSIN DD statement or by allocating it to DUMMY. Figure 32 shows the sample JCL used to run RACF remove ID utility.

```
//USER01   JOB  Job card...
//CLEANUP  EXEC PGM=IRRRID00,REGION=25M
//SYSPRINT DD   SYSOUT=*
//SYSOUT   DD   SYSOUT=*
//SORTOUT  DD   UNIT=SYSALLDA,SPACE=(CYL,(5,5))
//SYSUT1   DD   UNIT=SYSALLDA,SPACE=(CYL,(3,5))
//INDD     DD   DISP=OLD,DSN=USER01.IRRDBU00.DATA
//OUTDD    DD   DISP=OLD,DSN=USER01.IRRRID00.CLIST
//SYSIN    DD   DUMMY
/*
```

Figure 32. Searching for All Residual References

Searching for a List of IDs

IRRRID00 can be used to find specific user IDs and group names. Figure 33 shows the sample JCL used to run RACF remove ID utility and search for the IDs MARK, BRUCE, and JUNO.

```
//USER01   JOB  Job card...
//CLEANUP  EXEC PGM=IRRRID00,REGION=25M
//SYSPRINT DD   SYSOUT=*
//SYSOUT   DD   SYSOUT=*
//SORTOUT  DD   UNIT=SYSALLDA,SPACE=(CYL,(5,5))
//SYSUT1   DD   UNIT=SYSALLDA,SPACE=(CYL,(3,5))
//INDD     DD   DISP=OLD,DSN=USER01.IRRDBU00.DATA
//OUTDD    DD   DISP=OLD,DSN=USER01.IRRRID00.CLIST
//SYSIN    DD   *
MARK
BRUCE
JUNO
/*
```

Figure 33. Searching for Specific References

Specifying a Replacement ID

Figure 34 on page 347 shows the sample JCL used to run the RACF remove ID utility and search for the IDs MARK, BRUCE, and JUNO, with a replacement ID for MARK.

```
//USER01   JOB  Job card...
//CLEANUP  EXEC PGM=IRRRID00,REGION=25M
//SYSPRINT DD   SYSOUT=*
//SYSOUT   DD   SYSOUT=*
//SORTOUT  DD   UNIT=SYSALLDA,SPACE=(CYL,(5,5))
//SYSUT1   DD   UNIT=SYSALLDA,SPACE=(CYL,(3,5))
//INDD     DD   DISP=OLD,DSN=USER01.IRRDBU00.DATA
//OUTDD    DD   DISP=OLD,DSN=USER01.IRRRID00.CLIST
//SYSIN    DD   *
MARK ELVIS
BRUCE
JUNO
/*
```

Figure 34. Specifying a Replacement ID

Finding Residual IDs

The remove ID utility searches these fields for any references to user IDs or group names that do not exist as a user profile or a group profile:

- Owner
- Superior group
- Subgroup
- Default group
- Connections
- Connection owner
- Notify
- Standard access list
- Conditional access list
- DFP(RESOWNER)
- STUSER
- STGROUP
- GROUPS field of the TME segment for general resource profiles in the ROLE class
- APPLDATA field of general resource profiles in the following classes:
  - DCEUUIDS
  - DIGTCERT
  - DIGTCRIT
  - DIGTNMAP
  - KERBLINK
  - NDSLINK
  - NOTELINK
  - TMEADMIN

  Note: RDELETE commands created for profiles in the following classes may not be executed successfully if the profile name contains lowercase characters:
  - KERBLINK
  - NDSLINK
  - NOTELINK

  See *z/OS SecureWay Security Server RACF System Programmer's Guide* for information about recovering from these failures.

- Data set high-level qualifier
- GLOBAL DATASET high-level qualifier
- Profile names of certain general resource member classes

A list of user IDs and group names is generated from this processing. This list of IDs is used to create the appropriate RACF commands.

Running IRRRID00 with an Empty SYSIN

You may want to run the remove ID utility regularly, as part of a RACF database cleanup task that you perform periodically, for example. When you run the utility with a `SYSIN DD DUMMY` statement, the output show all occurrences of residual IDs in your RACF database.

```
//CLEANUP  EXEC PGM=IRRRID00,REGION=25M
//SYSPRINT DD   SYSOUT=*
//SYSOUT   DD   SYSOUT=*
  .
  .
  .
//INDD    DD   ...
//OUTDD   DD   ...
//SYSIN   DD   DUMMY
```

Figure 35. Running IRRRID00 with an Empty SYSIN: Sample Input

```
PERMIT   'A.B.**'   ID(MARK) DELETE
PERMIT   'UPROC1'   CLASS(TSOPROC) ID(MARK) DELETE
PERMIT   '12345'    CLASS(ACCTNUM) ID(MARK) DELETE
PERMIT   'A.B.**'   ID(JUNO) DELETE
PERMIT   'A.B.**'   OWNER(?JUNO)
```

Figure 36. Running IRRRID00 with an Empty SYSIN: Sample Output

As shown in Figure 36, IRRRID00 found several references to MARK and JUNO, even though these users did not have a profile in the USER class. IRRRID00 produces the commands you would need to change or remove these references in the various fields.

Creating Commands to Remove IDs

A list of IDs from the SYSIN file or from the residual search processing is used to create the appropriate RACF commands. While creating the RACF commands, the remove ID utility searches these fields:

- All fields that were searched in finding residual IDs
- All data set qualifiers
- All general resource qualifiers
- Selected member data

  See “Processing General Resource Profiles” on page 353 for more information about general resources and member data.

Running IRRRID00 with Data in SYSIN

If you run the remove ID utility with *oldid newid* as input, *newid* replaces all references to *oldid* in these fields:

- DFLTGRP
- OWNER

- RESOWNER
- SUPGROUP

The *newid* value must follow the *oldid* value on the same input record, separated from it by at least 1 blank.

In this example, MARK was specified in SYSIN. IRRRID00 now produces only RACF commands relative to this user ID. In addition, when you run IRRRID00 with a list of IDs, delete commands (DELDSD or RDELETE) are created for all data set and general resource profiles that have one of the IDs as a qualifier. The search for IDs within profile names is done without regard to any generic character in the profile name.

```
//CLEANUP  EXEC PGM=IRRRID00,REGION=25M
//SYSPRINT DD   SYSOUT=*
//SYSOUT   DD   SYSOUT=*
  .
  .
  .
//INDD     DD   ...
//OUTDD    DD   ...
//SYSIN    DD   *
MARK
/*
```

Figure 37. Running IRRRID00 with Data in SYSIN: Sample Input

```
ALTDSD   'A.B.**'  OWNER(?MARK)
ALTDSD   'A.B.**'  NONOTIFY
ALTDSD   'A.B.**'  DFP(RESOWNER(?MARK))
PERMIT   'A.B.**'  ID(MARK) DELETE
PERMIT   'A.B.**'  WHEN(PROGRAM(ABCD)) ID(MARK) DELETE
```

Figure 38. Running IRRRID00 with Data in SYSIN: Sample Output

IRRRID00 Output

The output of the RACF remove ID utility is a set of commands that remove or alter the references to residual authority IDs or to IDs in the list of IDs specified in the SYSIN data set. References on an access list are deleted with the PERMIT command. For references requiring that another ID value be specified, commands are created to change the reference to another value.

When IRRRID00 searches for references, it creates delete commands (DELDSD or RDELETE) for all data set and general resource profiles that have one of the IDs as a qualifier. IRRRID00 searches for IDs within profile names without regard to any generic characters within the profile name.

In some cases, such as an ID in an access list or in a NOTIFY field, the command simply removes the ID. In other cases, such as an OWNER field, the ID cannot simply be removed. In these cases, IRRRID00 creates a value that is the replacement ID value. The default replacement ID value is **?***id*, where *id* is the ID that is being replaced. For example, if the utility was searching for the ID MARK, which is the owner of profile IRRDBU00.JCL.*, IRRRID00 generates this command:

```
ALTDSD DA('IRRDBU00.JCL.*') GENERIC OWNER(?MARK)
```

Because ?MARK is not a syntactically valid ID, you cannot run this command. Use the editor of your choice to change ?MARK to the ID you want to own this profile. If MARK owned more than one profile, you can globally change ?MARK to the new ID field for all of its occurrences with a global edit command. For example, to change all of MARK's ownership to ELVIS using ISPF, enter:

```
C ?MARK ELVIS ALL
```

```
/*************************************************************/
/*                                                           */
/* The RACF Remove ID Utility (IRRRID00) was executed on     */
/* 1998-03-23 at 09:00:01.                                   */
/*                                                           */
/* This file contains RACF commands that can be used to      */
/* identify references to user IDs and group names. Residual */
/* references on an access list are deleted with the PERMIT  */
/* command. For all other references, commands are created to */
/* change the reference to another value. The default value  */
/* is ?id. This allows all references to a particular ID to  */
/* be easily changed to another value using a text editor.   */
/*                                                           */
/* Commands to alter ROLE definitions will be created within */
/* comments for informational purposes, though the actual    */
/* updates should be made from Tivoli. The ROLE will not be  */
/* updated with a replacement value for a group name.        */
/*************************************************************/

/*************************************************************/
/* The INDD data set has been scanned for all names that do  */
/* not have a user or group name defined for them in INDD. This */
/* list of names has been formatted and sorted into the      */
/* SORTOUT data set.                                         */
/*************************************************************/

 CONNECT BILL   GROUP(RACFDEV )    OWNER(?ELVIS )
 ALTDSD  'DASDDEF.VCE313S' GENERIC OWNER(?JUNO  )
 PERMIT  D12*   CLASS(DASDVOL ) ID(MARK  ) DELETE
 PERMIT  111111 CLASS(DASDVOL ) ID(BRUCE ) DELETE
 PERMIT  222222 CLASS(DASDVOL ) ID(JUNO  ) DELETE
 RALTER  FACILITY IRR.LISTUSER NONOTIFY
 RALTER  FACILITY IRR.PASSWORD.RESET OWNER(?TERRY  )
/* RALTER  ROLE DEVELOPMENT TME(DELGROUPS(RACFDEV ))        */

/*************************************************************/
/* The following commands delete profiles. You must review   */
/* these commands, editing them if necessary, and then remove */
/* the EXIT statement to allow the execution of the commands. */
/*************************************************************/

 EXIT

 RDELETE  TMEADMIN TERRY@TWINPEAKS.COM
 DELDSD   'D69A.BRUCE.TEXT' VOLUME(TSO018) NOSET
 DELDSD   'D69A.MARK.*'
 DELUSER  BRUCE
 DELUSER  JUNO
 DELUSER  MARK
 DELGROUP RACFDEV

/*************************************************************/
/* IRRRID00 has successfully completed                       */
/*************************************************************/
```

Figure 39. Sample Output from the IRRRID00 Utility

Issuing the Commands Generated by IRRRID00

Before running the commands that IRRRID00 creates, you need to review the IRRRID00 output. IRRRID00 places the commands that alter existing profiles before the commands that delete profiles. IRRRID00 places an `EXIT` command between these two sets of commands. By making TSO stop the second set of commands

from running, EXIT prevents the deletion of profiles until after you have reviewed the commands generated by IRRRID00. You must remove the EXIT statement to run the commands that follow it.

Processing Profiles and Resources

IRRRID00 creates commands that change the protection of your resources. You should make sure that the resources protected by the RACF profiles that are being altered or deleted have been properly renamed, deleted, or protected by other RACF profiles.

A resource that was protected by a profile that IRRRID00 has deleted is now protected by another less specific profile, your installation's PROTECTALL value (for data sets only), or any installation exits.

When IRRRID00 generates a DELDSD command to remove a profile for a discrete data set, it uses the NOSET operand, which leaves the RACF-indicated bit on in the VTOC.

Any data sets that have a high-level qualifier (HLQ) of a user ID or a group name that no longer exists should be archived or assigned new high-level qualifiers. You should consider renaming the data sets to another HLQ to ensure that they have proper protection and ownership.

IBM's DFSMSdss component (or equivalent) can be used to delete or rename data sets. With appropriate profiles in the RACF FACILITY class, you can use the ADMIN option on DFSMSdss commands:

- COPY with delete and rename unconditional
- DUMP with delete followed by RESTORE with rename unconditional.

DFSMSdss also provides the following special patch-flags, which are effective only when ADMIN is used:

- Changing Default Protection Status During Restore

  Offset 13 Turns off the RACF indicator in the volume table of contents
- Bypass Storage and Management Class Authorization Checking During Restore

  Offset 16 Bypass failures due to the owner of the resource being a revoked user ID
- Bypass Storage and Management Class Authorization Checking During Copy[7]

  Offset 3C Bypasses failures due to the owner of the resource being a revoked user ID
- Allow COPY with DELETE of RACF Indicated Data Sets and No Discrete Profile[7]

  Offset 3D Requests a warning instead of an error condition

For more information on:

- DFSMSdss commands, see *z/OS DFSMSdss Storage Administration Reference*.
- The ADMIN option and the appropriate RACF FACILITY class profiles, see *z/OS DFSMSdss Storage Administration Reference*.
- The DFSMSdss patch-flags, see *z/OS DFSMSdss Diagnosis Guide*.

7. The patch-flags at offset 3C and 3D are recent additions to DFSMSdss 1.2. See the closing text in APARs OW09751 and OW10314, respectively, for more information.

What IRRRID00 Verifies

When doing a residual search for ID values that are no longer valid, IRRRID00 verifies that the ID value is correct in the context that it is used. For example, a NOTIFY field may only have a user ID as its value. If IRRRID00 determines that a NOTIFY field contains a group name, IRRRID00 then searches that IRRDBU00 output for all occurrences of that ID and creates commands to delete those occurrences, allowing you to ensure that the ID has the correct access authorities.

Database Objects That Are Not Processed

RACF requires that several key RACF objects always exists. IRRRID00 does not create commands that delete these items:

- the user IDs: IBMUSER, `irrcerta`, `irrmulti`, and `irrsitec`
- the group names: SYS1, VSAMDSET, and SYSCTLG
- connections between IBMUSER and SYS1, VSAMDSET, and SYSCTLG
- the superior group of VSAMDSET and SYSCTLG
- the SECLABEL profiles SYSLOW, SYSHIGH, and SYSNONE

Processing a Hierarchy of Groups

When IRRRID00 encounters a hierarchy of groups in which a group is a superior group to a second group and both of the groups are being deleted, IRRRID00 temporarily changes the superior group to SYS1 and deletes the groups.

Processing Global Profiles

GLOBAL data set profiles are processed like data set profiles. If no SYSIN data is specified, IRRRID00 searches the database looking for potential residual IDs. During this search, general resource profiles are not searched, with the exception of GLOBAL data set profiles. GLOBAL data set profile names are processed just like data set profile names: the HLQs of profiles are examined for residual IDs.

The special qualifier names that are allowed in a GLOBAL data profile, such as &RACUID and &RACGRP, are not considered residual IDs.

Processing General Resource Profiles

IRRRID00 does not search the names of general resource profiles that do not contain user ID or group name values. Classes in which profiles names are not searched for ID values include TAPEVOL, DASDVOL, RACGLIST, SECLABEL, and SECDATA classes. Member data in these classes or their GLOBAL equivalents is not searched.

The profile names of VMEVENT and VMXEVENT profiles are searched, but member data in these classes or their GLOBAL equivalents is not searched.

Processing MEMBER Data

IRRRID00 only processes the first 252 characters of data for an ADDMEM or DELMEM operand. Commands with truncated data are preceded by a question mark (**?**). VMEVENT/VMXEVENT member records are not processed.

Processing Universal Groups

If you wish to delete a universal group, you should run IRRRID00, specifying the group name, to delete the group and remove all member connections. The DELGROUP command may successfully delete a UNIVERSAL group that has members connected to it because universal group profiles may not contain all

connected user IDs in the member list. However, when you delete a universal group, you will receive the ICH05008I informational warning message that the group is a universal group, reminding you to run IRRRID00. For more information about universal groups, see “Defining Large Groups with the UNIVERSAL Attribute” on page 54.

Be sure to execute the resulting REMOVE commands to remove all users from the universal group. If you do not, the profiles of those users will contain residual data regarding the deleted group connection. You should also execute any resulting PERMIT DELETE commands to remove residual entries from access lists that contain the deleted universal group.

IRRRID00 and Tivoli

The RACF remove ID utility detects profiles in the TMEADMIN class when the input user ID is in the APPLDATA field of the TMEADMIN class profile.

IRRRID00 also finds occurrences of group names in the GROUPS field of the TME segment for general resource profiles in the ROLE class. You should make updates to ROLE profiles by changing the role definition from the Tivoli desktop and distributing the change to the z/OS or OS/390 system. The commands generated by IRRRID00 to remove the group references are commented out in the IRRRID00 output data set. If Tivoli has left a residual group reference in this field, you can uncomment the command and run the output EXEC.

If a replacement group name is specified in the SYSIN data set, IRRRID00 does not generate the command to add the new group name to the GROUPS field in the TME segment. Again, this is because the updates should be performed from the Tivoli desktop.

A change made locally to RACF does not have any effect on resource access due to role membership. If this change is not also made to the Tivoli database, the local RACF modification will be overridden the next time the role is distributed from Tivoli.

Time Required to Run IRRRID00

The amount of processing time IRRRID00 requires and how much I/O it performs depends on:

- The size of the database it is processing
- The number of IDs it is searching for
- The number of commands it will create
- Whether it is performing a residual search.

Periodically, IRRRID00 displays the number of IRRDBU00 records it has processed. The message numbers are IRR68019I and IRR68020I. These messages describe the number of records that have been searched (if a residual processing search was specified) and the number of records that have been processed looking for references to IDs.

Chapter 10. The RACF Remote Sharing Facility (RRSF)

This chapter describes aspects of the RACF remote sharing facility (RRSF) that security administrators should be aware of.

The RACF remote sharing facility allows RACF to communicate via APPC with other MVS systems that use RACF, allowing you to maintain remote RACF databases. RRSF extends the RACF operating environment beyond the traditional single host and shared DASD environments, to an environment made up of RRSF nodes that are capable of communicating with one another. This support provides administration of multiple RACF databases from anywhere in the RRSF network.

Benefits of RRSF support for the security administrator include:

- Administration from anywhere in the RRSF network.

  With RRSF, a security administrator logged on to one system in the RRSF network can direct allowed RACF TSO commands to remote RRSF nodes in the RRSF network. Administration of all the RACF systems in the RRSF network can take place from a single point of control.
- User ID associations.

  By supporting user ID associations and password synchronization, RRSF gives users with multiple user IDs the option of keeping their user ID passwords automatically synchronized across multiple systems.
- Automatic synchronization of databases. With automatic direction, RACF can keep databases synchronized automatically. When a command or application updates a database, RACF can automatically make the change to other databases.

The RRSF Network

The *RRSF network,* the foundation of the RRSF environment, is the structure through which RRSF functions operate. An RRSF network is made up of RRSF nodes. An *RRSF node* is made up of one or more z/OS or OS/390 systems running with active RACF subsystems. Figure 40 on page 357 illustrates an RRSF network.

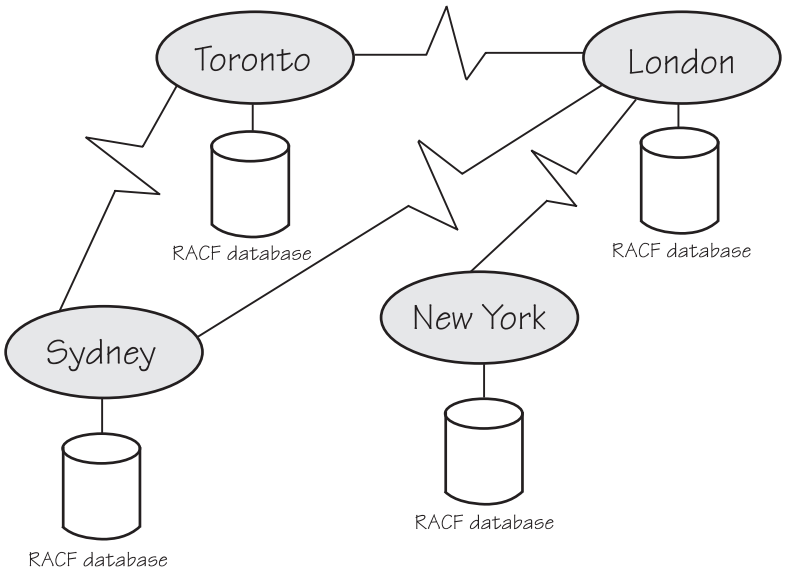


Figure 40. An RRSF Network

RRSF Nodes

An RRSF node is an MVS system image that has been defined as an RRSF node to RACF by a TARGET command. For information about defining RRSF nodes with the TARGET command, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

To direct commands or application updates from one MVS system image to another or synchronize passwords between users on two or more MVS system images, both system images must first be defined to RACF as RRSF nodes that can communicate with each other.

Local and Remote RRSF Nodes

In an RRSF network, the *local node* is the node whose viewpoint you are speaking from. Its *remote nodes* are the other nodes in the network the local node communicates with. For example, in the network shown in Figure 40:

- From the Toronto node's point of view, the Toronto node is the local node and the Sydney and London nodes are remote nodes. The Toronto node cannot communicate with the New York node.
- From the London node's point of view the London node is the local node and the Toronto, Sydney, and New York nodes are remote nodes.
- From the Sydney node's point of view, the Sydney node is the local node and the Toronto and London nodes are remote nodes. The Sydney node cannot communicate with the New York node.
- From the New York node's point of view, the New York node is the local node and the London node is a remote node. The New York node cannot communicate with the Sydney and Toronto nodes.

Single-System and Multisystem RRSF Nodes

In an RRSF network, each local or remote node can also be a *single-system node* or a *multisystem node*.

A *single-system RRSF node* is an RRSF node that consists of one MVS system image that does not share its RACF database. For example, in the network shown in Figure 40 on page 357, the Toronto node uses its own RACF database, which it shares with no other system.

A *multisystem RRSF node* is an RRSF node that consists of multiple MVS system images that share the same RACF database. One of the systems is designated as the main system and receives most of the RRSF communications sent to the node. For example, in the network shown in Figure 40 on page 357, the New York node might consist of two systems, Bronx and Brooklyn, that share a RACF database. One of the systems, Bronx, is designated as the main system. This does not affect its status as a local or remote node.

Local and Remote Modes

An RRSF node can operate in either local mode or remote mode.

When an RRSF node operates in *local mode*, it cannot communicate with other RRSF nodes. A node operating in local mode provides some remote sharing functions:

- Users with multiple user IDs on the node can synchronize passwords between those user IDs.
- Users with multiple user IDs on the node can direct commands to run under the other user IDs.
- Users can direct commands from their user IDs on the node to the same user ID. This allows you to run commands asynchronously in the RACF subsystem address space.

When an RRSF node operates in *remote mode*, it can communicate with other RRSF nodes. A node operating in remote mode provides all remote sharing features, so you can perform RACF functions across a network.

Establishing User ID Associations in the RRSF Network

Once the RRSF environment has been configured, users can establish associations (called user ID associations) between two RACF user IDs on nodes in the network. The user ID associations are used to:

- Link (associate) two user profiles on the same or different nodes
- Enable password synchronization to occur
  - Between a user's user IDs on the same node
  - Between a user's user IDs on different nodes.
- Allow a user to manually direct most RACF commands to execute on other user IDs with which the user's user ID has an appropriate association.

A RACF user ID can have multiple user ID associations. User ID associations can be set up by each user or by the security administrator using the RACLINK command. To enable the use of RACLINK, you need to create a profile in the RRSFDATA class. Once you've created the profile, you can restrict it to a subset of users. See "Controlling Access to the RACLINK Command" on page 282 for more information on enabling the use of the RACLINK command.

Types of User ID Associations

User ID associations can be either peer or managed associations.

A *peer user ID association* occurs between two user IDs and enables the user of either user ID to run allowed RACF commands under the authority of the other using 2-way command direction. Password synchronization is allowed in peer associations and these associations can be deleted by either user.

A *managed user ID association* enables the managing user ID to run allowed RACF commands under the authority of the managed user ID using 1-way command direction. Users of the managed user ID cannot run RACF commands under the authority of the managing user ID. Password synchronization is not allowed in managed associations and these associations can be deleted by either user.

Password Synchronization

With RRSF, users with multiple user IDs can keep their passwords synchronized across RACF databases. Password synchronization can be requested between user IDs when a peer user ID association is established with the RACLINK command and the PWSYNC option is specified.

The passwords of the user IDs do not need to be synchronized at the time the association is requested, nor are they synchronized when the association is established. The passwords are synchronized when either of the associated user IDs initiates a password change. The password history is updated with the old password on all systems where the password change occurs.

Password synchronization can occur for password changes initiated by:

- Logon processing
- The PASSWORD command
- The ALTUSER command
- Application programs that use the RACROUTE REQUEST=VERIFY macro to supply the user's new password in clear text form.
- Application programs that use the ICHEINTY or RACROUTE REQUEST=EXTRACT,TYPE=REPLACE to supply the user's new password in clear text form.

Note: Password changes initiated by the ADDUSER command do not result in password synchronization because the new user ID is not yet part of a user ID association.

The security administrator can enable or disable password synchronization for user IDs that have established a peer user ID association with password synchronization requested. See "Controlling Password Synchronization" on page 283 for more information.

Message Processing

Messages about the status of a password change and the password synchronization request can be viewed by editing the user's RRSFLIST data set, depending on the option specified on the SET PWSYNC command. See "Capturing Command Output" on page 365 for information about output handling for RACF commands that execute in the RACF subsystem address space. TSO end users may receive notification via the TSO SEND command that a password change request has been processed on their behalf by the command output handling components of RRSF.

The user who originates the password change is the source user, and the user to whom the source user directs a password change is the target user.

Messages issued by RRSF are returned as specified on the SET PWSYNC command.

The password history of the target user is updated with the user's old password by RRSF password synchronization support.

If the RACF database manager is unable to communicate a password change request to the RACF subsystem address space, message IRR417I is issued to the operator's console via WTO.

Output Capturing

Figure 41 shows captured output from a successful password synchronization request.

```
======================================================================
Password synchronization request issued at 15:03:58 on 04/28/98 was
processed at NODE1.TSOUSER on 04/28/98 at 15:04:00
REQUEST ISSUED: From user TSOUSR3 at NODE1 to user TSOUSER at NODE1.

 REQUEST OUTPUT:
 IRRC013I Password synchronized successfully for TSOUSR3 at NODE1 and
 TSOUSER at NODE1.
======================================================================
```

Figure 41. Captured Output From a Password Synchronization Request

The RACLINK Command

The RACLINK command is a RACF command that can be used to:

- Define user ID associations between pairs of user IDs
- Approve defined user ID associations
- Establish password synchronization between pairs of user IDs
- Delete pending or approved user ID associations
- View information about user ID associations

The general syntax of the RACLINK command is:

```
RACLINK ID(userid) DEFINE(node.userid/password) type
                   APPROVE(node.userid)
                   UNDEFINE(node.userid)
                   LIST(node.userid)
```

The operands you can specify with the RACLINK command are:

Operand	Function
ID(*userid***)**	Specifies one or more user IDs for whom the RACLINK operation is being performed. If this operand is not specified, the default is the user ID issuing the command.

DEFINE(*node.userid/password***)**
Associates (links) two user IDs, forming a user ID association.

type

- PEER(NOPWSYNC) (the default)
- PEER(PWSYNC)
- MANAGED

These types of user ID associations are:

- Peer association (with or without password synchronization)

- Managed association (password synchronization is not permitted).

Command direction is allowed in both types of user ID associations. In peer associations, command direction is 2-way. In managed associations, command direction is 1-way, from the managing user ID to the managed user ID.

The association is considered to be pending until an implicit or explicit approval of the association is made. While the association is pending, it can't be used for password synchronization or command direction.

Explicit approval of the association is needed unless one of the following is true:

1. The password of the target user ID is specified in the DEFINE operand
2. You are creating a user ID association and, for both profiles, one of the following is true:
   a. You have SPECIAL or group-SPECIAL authority
   b. You have a user ID association with a user who has this authority
   c. You are the owner of the profiles being changed.

Either user in a user ID association can delete the association, regardless of the type of association established.

Note: The security administrator can enable or disable the use of the DEFINE operand. See "Controlling the Use of the RACLINK DEFINE Operand" on page 282 for more information.

APPROVE Indicates approval of a pending user ID association. The association must be approved by the target of the DEFINE.

UNDEFINE Deletes a pending or approved user ID association. Either member of a user ID association can delete the association, regardless of the type of association established. All associations must be deleted before a user ID can be deleted.

LIST Provides the following information about the existing user ID associations for each of the user IDs specified.
- Association type (peer or managed)
- Password synchronization (yes, no, or not applicable)
- Association status (established or pending).

RACLINK can be issued as an MVS operator command or as a TSO command. It cannot be used as a directed command. For a complete description of the RACLINK command, see *z/OS SecureWay Security Server RACF Command Language Reference*.

User ID Associations

Using the RACLINK command, you can define, approve, delete, and list user ID associations.

RRSF

Defining User ID Associations

You can define these types of user ID associations:

- For your own user ID, with password synchronization
- For your own user ID, without password synchronization
- For other users, with password synchronization
- For other users, without password synchronization
- Managed user ID associations

Defining User ID Associations For Your Own User ID

To define a user ID association for your own user ID, the syntax of the RACLINK command is:

```
RACLINK DEFINE(node.userid/password) type
```

With Password Synchronization: To define a user ID association with password synchronization between your two user IDs, WILLIE on MVS01 and WONKA on MVS03, enter the following command from user ID WILLIE on MVS01:

```
RACLINK DEFINE(MVS03.WONKA) PEER(PWSYNC)
```

Without Password Synchronization: To define a user ID association without password synchronization between your two user IDs, WILLIE on MVS01 and WONKA on MVS03, enter the following command from user ID WILLIE on MVS01:

```
RACLINK DEFINE(MVS03.WONKA) PEER(NOPWSYNC)
```

Defining User ID Associations For Other Users

To define user ID associations for other users, the syntax of the RACLINK command is:

```
RACLINK ID(userid) DEFINE(node.userid/password) type
```

With Password Synchronization: To define a user ID association with password synchronization between VIOLET on MVS01 and VERUCA on MVS03, enter the following command on MVS01:

```
RACLINK ID(VIOLET) DEFINE(MVS03.VERUCA) PEER(PWSYNC)
```

Without Password Synchronization: To define a user ID association without password synchronization between VIOLET on MVS01 and VERUCA on MVS03, enter the following command on MVS01:

```
RACLINK ID(VIOLET) DEFINE(MVS03.VERUCA) PEER(NOPWSYNC)
```

Managed User ID Associations

To define a managed user ID association, the syntax of the RACLINK command is:

```
RACLINK DEFINE(node.userid/password) MANAGED
```

For example, to define a managed user ID association between VIOLET on MVS01 and VERUCA on MVS03, enter the following command on MVS01:

```
RACLINK ID(VIOLET) DEFINE(MVS03.VERUCA) MANAGED
```

Approving User ID Associations

To approve a pending user ID association, the syntax of the RACLINK command is:

```
RACLINK ID(userid) APPROVE(node.userid)
```

For example, to approve a pending user ID association between VIOLET on MVS01 and VERUCA on MVS03, enter the following command on MVS01:

```
RACLINK ID(VIOLET) APPROVE(MVS03.VERUCA)
```

Deleting User ID Associations

To reject a pending association or to delete an existing association, the syntax of the RACLINK command is:

```
RACLINK ID(userid) UNDEFINE(node.userid)
```

For example, to reject a pending user ID association between VIOLET on MVS01 and VERUCA on MVS03, enter the following command on MVS01:

```
RACLINK ID(VIOLET) UNDEFINE(MVS03.VERUCA)
```

Listing User ID Associations

To list user ID associations, the syntax of the RACLINK command is:

```
RACLINK ID(userid) LIST(node.userid)
```

The default is RACLINK LIST(*.*).

For example, to see all the associations defined for user ID CHARLIE, you could issue the following command:

```
RACLINK ID(CHARLIE) LIST(*.*)
```

and receive the following output:

```
ASSOCIATION information for user ID CHARLIE on node MVS01
at 9:27:12 on 04/18/98:

Association   Node.userid        Password  Association
  Type                           Sync      Status
------------  -----------------  --------  --------------

PEER OF       MVS01.VIOLET        YES      ESTABLISHED

PEER OF       MVS03.VERUCA        YES      ESTABLISHED

PEER OF       MVS01.WILLIE        NO       ESTABLISHED

MANAGER OF    MVS03.MIKE          N/A      ESTABLISHED

MANAGER OF    MVS03.AGLOOP        N/A      ESTABLISHED
```

Figure 42. RACLINK ID(userid) LIST(.*) Output*

Command Direction

Command direction can be used to perform security administration for multiple data centers from a central site without submitting batch jobs or logging on to the remote systems when the AT option is specified. Command direction using the AT option is not usually used when the remote databases are kept synchronized. In that case, automatic direction is used. Most RACF TSO commands can be manually directed on the local node or to a remote node within an RRSF network.

Manual command direction extends the user's RACF TSO command execution environment to include any of the RRSF nodes defined as targets of the node the user is logged on to, provided the user has an associated user ID on the target node and the associated user ID has the appropriate authority to run the command.

You can enable the use of command direction by creating a profile in the RRSFDATA class. You can restrict the use of command direction by not creating the

profile (so no one can direct commands) or by creating a profile and restricting access to it. See “Controlling the Use of the AT Operand” on page 284 for more information on restricting the use of the AT option.

Commands That Are Not Eligible for Command Direction

The following commands are not eligible for command direction:

- BLKUPD
- RACDCERT
- RACLINK
- RACF operator commands
- RACF TSO commands, when they are issued as operator commands

Directing Commands Using the AT Option

Once a peer user ID association is established, either user in the association can use the AT option to direct allowed RACF commands to run under the other user's authority. The commands run in the RACF subsystem address space at the other user's node.

The user specifies as the target node a node in an association and a node the user is allowed to direct to via RRSFDATA profiles. Commands can be directed only to a node with which the user has a RACLINK association. In addition, the user must have access to the DIRECT.nodename profile. If this is not true, the command cannot be directed unless the commands can be directed to your own ID on the local node only without any RACLINK association.

The target user ID specified in the AT option becomes the user ID that RRSF uses to determine if the requested command can be executed. That is, the user ID effectively becomes the command issuer at the target node and RRSF checks to see if that user ID has the proper authority to run the requested command.

When the command arrives, RRSF creates a subtask in the RACF subsystem address space for the specified user ID and performs authority checking while processing the requested command.

RACF TSO commands that specify command direction run asynchronously, that is, the command issuer does not wait until the command completes processing, and the command output is *not* automatically displayed at the command issuer's terminal. When the command completes processing, the command issuer may receive a TSO SEND message.

Any command output created via the PUTLINE service is captured by RRSF and saved in the issuing user's RRSFLIST data set.

Directing Commands on the Local Node

If a request specifies that it is to be handled by the local node, it runs in the RACF subsystem address space on the MVS system the request originates from.

Suppose you are USER2 on NODEC. To make RRSF operating in the RACF subsystem address space on NODEC process your LISTUSER request and run it from within the RACF address space, enter:

```
LISTUSER USER2 AT(NODEC.USER2)
```

This request makes RRSF start a subtask in the NODEC RACF subsystem address space for USER2 and invoke the LISTUSER command processor. The output is captured and put into a data set as described in “Capturing Command Output”.

If the target user ID on the local node is the same, no user ID association is needed. You only need a user ID association if the target user ID is different from the issuer’s user ID when only one system is involved. Also, you need authority to the DIRECT.*nodename* RRSFDATA profile for the local node.

Directing Commands On a Remote Node

If a request specifies that it is to be handled by a remote node, the local node sends the request to the target node specified by the AT value. The request is handled by the RRSF at that remote node.

For example, USER1 on NODED could direct a RACF TSO command to run in either the RACF subsystem address space on NODED under his own user ID authority, or direct the request to NODEC to run under the authority of USER2 (because USER1 on NODED has an approved user ID association with USER2 on NODEC.)

The request to have a LISTUSER for USER2 on NODEC issued by USER1 on NODED would look like:

```
LISTUSER USER2 AT(NODEC.USER2)
```

This command causes RRSF on NODED to send the request to NODEC. RRSF running in the RACF subsystem address space on NODEC creates a subtask for USER2 in the RACF subsystem address space and runs the LISTUSER command. The output is captured and sent back to NODED, where RRSF places it into USER1’s RRSFLIST data set, as specified in “Capturing Command Output”.

Capturing Command Output

When you direct a command, the results are returned to you and are appended to the bottom of your RRSFLIST user data set. If you do not have a RRSFLIST user data set, RRSF allocates one and adds the results.

The RRSFLIST user data set name is made up of the user’s prefix as specified by the user via the TSO PROFILE command, the user ID, and RRSFLIST. When the prefix and the user ID are the same, the duplicate qualifier is dropped. Thus, the data set name would be either *prefix.userid*.RRSFLIST or *userid*.RRSFLIST.

You will receive a TSO SEND message when the results are ready for viewing, for example:

```
LISTUSER was successful at node NODEC. Output written to USER2.RRSFLIST.
```

You do not receive a TSO SEND message if you had the TSO PROFILE NOINTERCOM setting in effect when you directed the command.

Users are responsible for maintaining their own RRSFLIST data sets. If a user’s data set becomes full, RRSF uses TSO TRANSMIT to send the command output to the user. The output begins with a message indicating that the user’s RRSFLIST data set was full at the time the output was received.

The contents of the data captured and appended to the RRSFLIST data set varies, but generally it contains:

- A brief description or summary of the event

- A reproduction, but not necessarily an exact replica, of the command issued. Command options that are not specified but defaulted by RACF may be included: security-sensitive data such as passwords or key codes are suppressed.

  The command is reproduced up to 255 characters, including the command options defaulted by RACF, and is truncated at this point. If it is truncated, the last three characters are replaced by a set of ellipses (″...″) to indicate that the remaining letters or options of the command had been omitted.
- The output produced by the command. This output is truncated after 4096 lines.

The following examples show the format of the captured output produced by commands running in the RACF subsystem address space. The format of the output shown is the same for both the user's RRSFLIST data set, and the TRANSMIT issued when the user's data set is full. Figure 43 shows the format of captured output for a directed LISTGRP command. Figure 44 shows the format of captured output for a directed ADDSD command.

```
 ================================================================================
 LG issued at 07:50:39 on 04/21/98 was processed at MVS03.SMITHJ on
 04/21/98 at 07:50:41

  COMMAND ISSUED: LG        (SYS1)

  COMMAND OUTPUT:
  INFORMATION FOR GROUP SYS1
      SUPERIOR GROUP=NONE         OWNER=SMITHJ
      NO INSTALLATION DATA
      NO MODEL DATA SET
      TERMUACC
      SUBGROUP(S)=SYSCTLG VSAMDSET
      USER(S)=     ACCESS=     ACCESS COUNT=          UNIVERSAL ACCESS=
        IBMUSER      JOIN        000017                    READ
           CONNECT ATTRIBUTES=NONE
           REVOKE DATE=NONE               RESUME DATE=NONE
================================================================================
```

Figure 43. Captured Output from a Directed LISTGRP Command

```
================================================================================
ADDSD issued at 11:34:29 on 04/21/98 was processed at MVS02.JWS on
04/21/98 at 11:34:31

 COMMAND ISSUED: ADDSD        'JWS.DEV*'

 COMMAND OUTPUT:
 IRRR008I Command succeeded. There are no messages.
================================================================================
```

Figure 44. Captured Output from a Directed ADDSD Command

All time stamps shown in the RRSFLIST data set are initially recorded as Greenwich Mean Time (GMT). These time stamps are meant to show the relative sequence in which the commands were entered and processed. When output or notify information is written into the RRSFLIST data set, these times are converted from GMT into local times. The time stamps are as accurate as possible, but they are not intended to give the exact, precise times of events. In addition, the accuracy of the time stamps depends on how accurately you have set your system clocks.

Note: RRSF assumes that either all nodes in the RRSF network have their clocks set to GMT and have appropriate local time offsets in SYS1.PARMLIB, or

that all nodes have their clock set to local time in the same time zone. Any other configuration will cause errors in the timestamps shown in an RRSFLIST data set.

Directing Commands Using the ONLYAT Option

The following information pertains only to automatic command direction.

Because automatic command direction provides a facility to keep RACF database profiles synchronized between RRSF nodes with respect to RACF TSO commands, there may be a need to fix a situation that has caused the RACF profiles to become out-of-synch. The ONLYAT option addresses this situation.

The ONLYAT option is restricted to SPECIAL users because it can potentially cause out-of-synch conditions if used improperly. It is a mechanism to direct RACF TSO commands to the same or other nodes in the same manner as the AT option, except that the command is not automatically directed. That is, it runs only on the node it is directed to. The command is processed in the RACF subsystem address space under the authority of the specified user ID provided the following requirements are met:

- Both the command issuer and the target user ID must be SPECIAL.
- If the target user ID is the same as the command issuer (although nodes can be different), no user ID association is required.
- If the target user ID is different from the command issuer, a user ID association between command issuer and target user ID is required. (This prevents a SPECIAL user from unauthorized use of another remote SPECIAL user ID.)

Automatic Direction

Automatic direction is an extension of command direction and password synchronization that allows some administrative tasks and application updates to be automated between RRSF nodes. Automatic direction keeps already synchronized RACF profiles synchronized between two or more remote nodes.

Automatic direction includes:

- **Automatic direction of commands**, which allows RACF TSO commands that update the RACF database to be automatically directed to remote nodes in order to keep profiles synchronized between the nodes. Commands issued with automatic direction of commands run asynchronously. The results and output from the commands are returned to specified users (not necessarily the command issuer).
- **Automatic password direction**, which keeps already-synchronized RACF user profiles synchronized between two nodes, with respect to RACF passwords.
- **Automatic direction of application updates**, which allows updates made by RACF macros to be propagated to the RACF databases of other systems. Updates to the RACF database can be made using:
  - ICHEINTY
  - RACDEF
  - RACROUTE REQUEST=DEFINE
  - RACROUTE REQUEST=EXTRACT,TYPE=REPLACE
  - RACXTRT

  Automatic direction of application updates allows these changes to be automatically sent to selected remote nodes. These updates to remote target

nodes take place only after the update has successfully completed on the local node where it is executing and the macro completes with a return code of 0.

Note: RACF database updates made by the RACDCERT command are candidates for propagation under the control of automatic direction of application updates, even though the RACDCERT command itself is not eligible for automatic command direction. In this case, the individual updates made by RACDCERT may be successful, and propagated to other nodes, even though the RACDCERT command as a whole may fail.

The essential elements of automatic direction are:

- Activation and deactivation, which are done with SET command options. See "Preparing to Use Automatic Direction" on page 369 and *z/OS SecureWay Security Server RACF Command Language Reference* for more information.
- Controlling which updates get automatically directed to which nodes.

  Application updates, password changes, and commands can be controlled by RRSFDATA profiles.
- Notification of appropriate users of results and output from automatically directed updates.

  An installation must decide who should be notified of results and output from automatically directed commands, application updates, and passwords. This is done with the SET command options OUTPUT and NOTIFY. See "Output Processing" on page 371 and *z/OS SecureWay Security Server RACF Command Language Reference* for more information.

Example: Automatic Direction of Commands

Automatic direction of commands works as follows: Suppose NODEA, NODEB, and NODEC have equivalent profiles in the USER and GROUP classes. All RACF TSO commands that affect USER and GROUP profiles can be automatically directed between the nodes. When an ADDUSER command is issued on NODEA, it can be automatically directed to execute on NODEB and NODEC. When a DELGROUP command is issued on NODEC, it can be automatically directed to NODEB and NODEA.

There might also be special situations where automatic direction can be used to facilitate administrative updates to multiple RACF databases. For example, suppose a university has a production MVS system and a test MVS system, each with its own RACF database. At the beginning of each semester, each new student gets a user ID on each MVS system. By temporarily using automatic direction of commands, an administrator can enter ADDUSER commands on the production system and have them automatically directed to the test MVS system.

Example: Automatic Password Direction

Automatic password direction does the following: NODEA, NODEB, and NODEC have equivalent profiles in the USER class. Password changes can be automatically directed to RACF user profiles without the need for established RACLINK PEER PWSYNC associations. When a user's password changes on NODEA, a password synchronization request can be automatically directed to execute on NODEB and NODEC.

Example: Automatic Direction of Application Updates

Automatic direction of application updates does the following: An installation-written application that creates profiles in installation-defined class using RACROUTE REQUEST=DEFINE can execute on NODEA and cause profiles to be created on NODEB and NODEC as well as NODEA.

Preparing to Use Automatic Direction

1. Check your RACF exits, naming conventions table, templates, and dynamic parse tables.

   If you are keeping profiles synchronized between multiple nodes, the above items on these nodes should be examined to see if they could cause out-of-synch conditions. See *z/OS SecureWay Security Server RACF System Programmer's Guide* for more information.

2. Decide which updates to automatically direct and which profiles to keep synchronized. Updates can be made by command, password changes, and applications. Decide which ones you want to send to target nodes.

   There are many considerations to synchronizing profiles because many profiles in the RACF database are related in important ways. An installation's goal should be to have RACF updates execute with the same results on each node (to minimize error conditions and out-of-synch conditions). Here are some recommendations to help prevent out-of-synch conditions:

   a. Use automatic direction to keep USER and GROUP profiles synchronized:
      - The same user IDs and groups need to exist on each node, because updates are automatically directed to the same user IDs.
      - The RACF authorities for each user ID authorized to automatically direct updates on a node should be equivalent to that same user ID's authorities on each node. This includes user authorities (such as SPECIAL and CLAUTH) and group authorities (which users are connected to which groups).

        For example, if SYSPROG on NODEA is SPECIAL and automatically directs commands to SYSPROG on NODEB, SYSPROG on NODEB should have SPECIAL authority also.
      - For automatic direction of commands:
        – If commands such as PERMIT or other commands that specify a group name or a user ID as an operand are being kept synchronized, the specified group names or user IDs must exist on both nodes.
        – If the CONNECT command is automatically directed, the GROUP class should be kept synchronized. Otherwise, the automatically directed CONNECT commands are likely to fail.
      - If the USER class is being kept synchronized, it should be noted that the RACLINK command is not eligible for automatic direction. Because the RACLINK command adds information to the USER profiles, an installation should consider whether to manually establish the user ID associations on each node; otherwise the USER profiles do not remain synchronized.

   b. Synchronize profiles by class rather than by command:
      - If RDEFINE DASDVOL is automatically directed and RALTER DASDVOL is not, the DASDVOL profile becomes out-of-synch.
      - If ADDSD is automatically directed, but PERMIT DATASET is not, the DATASET profile becomes out-of-synch.
      - If RDEFINE TAPEVOL is automatically directed, but application updates for the TAPEVOL class are not, the TAPEVOL profiles become out-of-synch.

c. Keep SETROPTS command settings synchronized:
   - This can be done manually (using command direction) if SETROPTS is issued infrequently.
   - Make sure that general resource classes on both nodes are active if updates are being automatically directed for the class.

d. Considerations for running the IRRRID00 utility:

   Because the DELUSER and DELGROUP commands do not automatically delete access list entries from resource profiles, you can use IRRRID00 to generate commands to clean up the profiles for a user or group that has been deleted. If the PERMIT command is being automatically directed, the user who runs the generated list of commands from IRRRID00 must be authorized to automatically direct the PERMIT commands. Otherwise, the resource profiles on the remote system will not be cleaned up.

e. When synchronizing a particular kind of profile between nodes, it is better to give all users who can update the profiles controlling those profiles the authority to direct updates automatically. To do this, you can give them READ authority to the appropriate RRSFDATA profiles, for example.

   If there are users who can cause updates to occur locally but cannot direct them automatically, some updates are not automatically directed, and the profiles do not remain synchronized. There are special considerations for automatic direction of application updates for the DATASET class. For these, see “Considerations for the DATASET Class” on page 382.

3. Synchronize specified profiles:
   - Run IRRRID00 on each RACF database to remove references to user IDs and group names that no longer exist.
   - Synchronize databases. You can do this by manually copying the database from one system to the other, or by using a tool that can help you synchronize databases. See “Internet Sources” on page xxv to find out how you can get access to this tool.
4. Create AUTODIRECT profiles in the RRSFDATA class as desired to specify which updates should be automatically directed. Remember to create the profiles on each node from which automatic direction should occur. For example, the following commands cause all password updates, commands, and application updates for the user class to be automatically directed to all remote nodes:

   ```
   SETROPTS GENERIC(RRSFDATA) (required if you use generic profiles)

   RDEFINE RRSFDATA AUTODIRECT.*.USER.* UACC(READ)
   ```

   For automatic password direction only, RDEFINE is:

   ```
   RDEFINE RRSFDATA AUTODIRECT.*.USER.PWSYNC UACC(READ)
   ```

   Note: The RRSFDATA profile PWSYNC.*nodename* is not required for using automatic password direction.
5. Activate the RRSFDATA class on each node, if it has not already been activated. It is recommended that the class be RACLISTed for performance reasons. If the class has already been RACLISTed, refresh the profiles. For example, use one of the following commands:

   ```
   SETROPTS CLASSACT(RRSFDATA) RACLIST(RRSFDATA)
   SETROPTS RACLIST(RRSFDATA) REFRESH
   ```
6. Decide who should be notified of the automatic direction results and output.
7. Activate automatic direction when you are ready by issuing the SET command with the options corresponding to automatic command direction, automatic

password direction, and automatic direction of application updates, with output and notification options appropriate to your installation's needs:

```
SET AUTODIRECT(...)
SET AUTOPWD(...)
SET AUTOAPPL(...)
```

Note: IBM recommends the use of a RACF parameter library as specified for RACF subsystem initialization. The SET command should be in the default IRROPTxx member of this library so that these options are reinstated when the subsystem is restarted or the entire system is reIPLed.

8. At a later date, to temporarily or permanently deactivate one or more automatic direction functions, issue the SET command with the options corresponding to automatic command direction, automatic password direction, and automatic direction of application updates:

   ```
   SET NOAUTODIRECT
   SET NOAUTOPWD
   SET NOAUTOAPPL
   ```

   See the note in step 7 on page 370 regarding the RACF parameter library.

9. The possibility of failing while attempting to execute a command issued on one (up-level) system and manually or automatically directed to another (down-level) system through RACF remote sharing has been present since the introduction of RACF 2.2. This can occur if the command references a class unknown to the target system (class descriptor tables are different), if it references a segment or field unknown to the target system (templates or dynamic parse definition are different), if it uses a command keyword unknown to the target (dynamic parse definitions or command processor code is different), or if it specifies a profile or member name that is unacceptable to the target system (class descriptor tables have different syntax requirements for profile name length or syntax).

   If an out-of-synchronization condition occurs while using automatic command direction, a RACF TSO command can be directed with the ONLYAT option to fix the condition. The command runs on the node specified on the ONLYAT option and are propagated to any other node. (Note that if the AT keyword is used, the command can be propagated by automatic command direction to other nodes.) For information on the ONLYAT option, see "Directing Commands Using the ONLYAT Option" on page 367. For a complete list of RACF commands that are eligible for automatic command direction, see *z/OS SecureWay Security Server RACF Command Language Reference*.

Output Processing

For automatic direction, the installation has many options concerning where the output and notification information resulting from an RRSF request may be sent. This flexibility is provided by operands of the SET command. For example, some installations might choose to notify only one or two administrators when errors are encountered during automatic direction.

The OUTPUT operand specifies that the output from automatic direction should be put in the RRSFLIST data set for the specified users. If the output cannot be put in the RRSFLIST data set for some reason, the output is transmitted to the users. The output usually contains messages issued during execution, such as informational, warning, or error messages. After RACF determines the result of execution, it sends back either a message saying it succeeded or a message giving diagnostic information.

The NOTIFY operand specifies that TSO SEND commands are issued to the specified users with the results of automatically directed updates. The information sent indicates whether the update was successful or unsuccessful, but does not include other details about the execution.

On both the OUTPUT and NOTIFY operands, you can specify whether the output or messages should be sent for all or some updates. The ALWAYS option specifies that results or output from all automatically directed updates are returned to the specified users. This should be used if the users are interested in the results of *every* automatically directed update.

Output includes informational, warning, and error messages. For automatic direction of commands:

- **WARN** specifies that results or output from an automatically directed update are returned to the specified users only when the return code from the update is at least four. This should be used if the users are interested in those automatically directed updates that complete with error conditions or warning conditions.
- **FAIL** specifies that results or output from an automatically directed update be returned to the specified users only when the return code from the update is at least 8. This should be used if the users are interested in only those automatically directed updates which complete with error conditions.

For automatic direction of application updates and passwords:

- WARN and FAIL have the same meaning. Either WARN or FAIL result in returned output or notification when a directed application update or password fails to completely take effect on a remote node.

If an automatically directed update error occurs, the RACF profiles become out-of-synch. An installation should specify at least one person (probably an administrator who is familiar with required profiles) on the OUTPUT and NOTIFY options to receive error results and output (FAIL option). If the initial values of NOOUTPUT and NONOTIFY are used, no one is notified when automatic direction errors occur. Also, once an update is automatically directed to a remote node, all output and messages associated with that update are discarded.

For more information and examples on using the SET command to specify options for automatic direction of updates, see *z/OS SecureWay Security Server RACF Command Language Reference*.

Effects of Using OUTPUT and NOTIFY

After the OUTPUT and NOTIFY operands have been specified on the SET AUTODIRECT command, they are used in the processing for automatic direction in the following ways.

Suppose for the examples that the following commands are in effect on NODE1:

```
SET AUTODIRECT (OUTPUT(WARN(NODE1.ANN)) NOTIFY(FAIL(NODE1.ANN)))
SET AUTOPWD    (OUTPUT(WARN(NODE1.ANN)) NOTIFY(FAIL(NODE1.ANN)))
SET AUTOAPPL   (OUTPUT(WARN(NODE1.ANN)) NOTIFY(FAIL(NODE1.ANN)))
```

and the following commands are in effect on NODE2:

```
SET AUTODIRECT (OUTPUT(WARN(NODE2.SAM)) NOTIFY(FAIL(NODE2.SAM)))
SET AUTOPWD    (OUTPUT(WARN(NODE2.SAM)) NOTIFY(FAIL(NODE2.SAM)))
SET AUTOAPPL   (OUTPUT(WARN(NODE2.SAM)) NOTIFY(FAIL(NODE2.SAM)))
```

1. When an update has been automatically directed to a remote node and execution is completed, the OUTPUT and NOTIFY settings are used on the

node at which the automatically directed update executes. The OUTPUT and NOTIFY settings on the node at which the update originally executed are not relevant.

For example, a user on NODE1 runs an application that performs an update, and the update is automatically directed to NODE2. On NODE2, the update runs with a return code of 8. SAM on NODE2 gets a NOTIFY message and the OUTPUT containing the error messages. Even though the update originally executed on NODE1, the OUTPUT and NOTIFY settings on NODE2 are used instead of the OUTPUT and NOTIFY settings on NODE1.

The update has successfully executed on one node and is automatically directed to another node. The NOTIFY and OUTPUT settings on that second node determine who is notified of the update completion.

2. When an error occurs while attempting to automatically direct an update to a remote node, the OUTPUT and NOTIFY settings are used on the node at which the update originally executes.

   For example, a user on NODE1 runs an application that performs an update, and the update is to be automatically directed to NODE2. As the update is placed in the OUTMSG workspace data set (which contains work items to be sent to NODE2), the data set becomes full; therefore, the update is not automatically directed to NODE2. ANN on NODE1 gets a NOTIFY message and the OUTPUT containing the error messages.

   The update has successfully executed on one node and is to be automatically directed to another node. An error occurs before the request arrives at the other node. The NOTIFY and OUTPUT settings on the first node determine who is notified of the error. This is treated as an error case (like an update which executed with return code 8), so any user with the FAIL, WARN, or ALWAYS setting would be notified or would receive output.

Recommendations for Use of OUTPUT and NOTIFY

To make sure that the appropriate users are notified or receive output, specify the same users on the OUTPUT and NOTIFY operands on the SET command issued on each node with automatic direction active.

The OUTPUT and NOTIFY lists should include users from at least 2 different nodes, if possible. This is important in the case of a workspace data set problem. For example, suppose a command executes successfully on NODE1 but cannot be sent to NODE2. If all the NOTIFY and OUTPUT users are also on NODE2, it is possible that RRSF might experience the same problem trying to notify the users as it experienced trying to send the command. By also specifying a user on NODE1 or by specifying one local user on each node to get the output, you ensure that RRSF can find someone to notify.

Output Data Set Names for Automatic Direction

When output from automatic direction is returned to the command issuer (&RACUID was specified on the OUTPUT operand), the RRSFLIST data set is allocated exactly as it is for directed commands—either *prefix.userid*.RRSFLIST or *userid*.RRSFLIST. The user's prefix is the one in effect at the time the RACF update was originally issued (as specified by the user via the TSO PROFILE command). It is recommended that application update output *not* be sent to &RACUID.

When output from automatic direction is returned to a user other than the issuer, the output is placed into a data set named either *prefix.userid*.RRSFLIST or *userid*.RRSFLIST. However, the prefix used is taken from the user's TSO segment at the time the output is returned; this prefix is the same as the prefix specified by

the user via the TSO PROFILE command except if the user is logged on when the output is received and the user has changed his or her TSO prefix during that logon session.

If the automatically directed command originated from the operator's console (even if &RACUID is specified on the OUTPUT operand), the TSO prefix is also taken from the user's TSO segment at the time the output is returned. The same is true for automatic direction of application updates that originate from batch jobs, started procedures, and other non-TSO environments.

Application update output should not be sent to &RACUID.

Notify Messages for Automatic Direction

The TSO SEND messages for automatic direction are the same messages used for command direction.

If the NOTIFY operand specifies &RACUID, and the command issuer has the TSO PROFILE NOINTERCOM setting in effect at the time the command is issued, the command issuer does not receive a TSO SEND message. This is similar to processing for a directed command.

For automatic direction of application updates the messages are similar except that they substitute, for example, ″Application update has completed successfully...″ for ″Command succeeded″.

Sample Output From Automatic Direction

Some sample output from automatic command direction follows:

```
=============================================================================
ADDUSER issued at 10:42:33 on 4/1/98 was processed at NODE1.LAURIE
on 4/1/98 at 10:43:45
Command was propagated by automatic direction from NODE2.LAURIE

 COMMAND ISSUED:  ADDUSER (ANDREW) PASSWORD()
 NAME('####################') AUTHORITY(USE) NOSPECIAL UACC(NONE)
 NOOPERATIONS NOADSP NOGRPACC NOAUDIT

 COMMAND OUTPUT:
 IRRR008I Command succeeded. There are no messages.
=============================================================================

=============================================================================
RDEFINE issued at 12:33:41 on 4/1/98 was processed at NODE1.LAURIE
on 4/1/98 at 12:35:02
Command was propagated by automatic direction from NODE2.LAURIE

 COMMAND ISSUED:  RDEFINE RRSFDATA AUTODIRECT.**  UACC(NONE)

 COMMAND OUTPUT:
 ICH10102I AUTODIRECT.** ALREADY DEFINED TO CLASS RRSFDATA.
=============================================================================
```

When errors occur during automatic direction of commands, the command output appropriately reflects what happened.

If an unexpected error occurred while trying to automatically direct a command to a remote node, the output may be similar to the following:

```
=============================================================================
RDEFINE issued at 12:33:41 on 4/1/98 was *not* processed at NODEA.LAURIE
Command was *not* propagated by automatic direction from NODEB.LAURIE
```

```
 COMMAND ISSUED:  RDEFINE RRSFDATA AUTODIRECT.**  UACC(NONE)

 ERROR INFORMATION:
 IRRR016I Command was not sent. Processing code is 502.
================================================================================
```

If an unexpected error occurs after an automatically directed command arrives at a remote node, the output may look as follows:

```
================================================================================
RDEFINE issued at 12:33:41 on 4/1/98 was *not* processed at NODEA.LAURIE
Command was propagated by automatic direction from NODEB.LAURIE

 COMMAND ISSUED:  RDEFINE RRSFDATA AUTODIRECT.**  UACC(NONE)

 ERROR INFORMATION:
 IRRC010I UNABLE TO ESTABLISH RACF ENVIRONMENT FOR COMMAND RDEFINE.
 IRRC012I TARGET USER ID NODEA.LAURIE DOES NOT EXIST.
================================================================================
```

All time stamps shown in the RRSFLIST data set are initially recorded as Greenwich Mean Time (GMT). These time stamps are meant to show the relative sequence in which the commands were entered and processed. When output or notify information is written into the RRSFLIST data set, these times are converted from GMT into local times. The time stamps are as accurate as possible, but they are not intended to give the exact, precise times of events. In addition, the accuracy of the time stamps depends on how accurately you have set your system clocks.

Note: RRSF assumes that either all nodes in the RRSF network have their clocks set to GMT and have appropriate local time offsets in SYS1.PARMLIB, or that all nodes have their clock set to local time in the same time zone. Any other configuration will cause errors in the timestamps shown in an RRSFLIST data set.

Some sample output from automatic direction of application updates follows.

Sample output for a successful RACDEF request:

```
================================================================================
Application update request issued at 15:42:33 on 4/1/98
was processed at NODE2.RACFU01
on 4/1/98 at 15:43:45
Request was propagated by automatic direction from NODE1.RACFU01

 REQUEST ISSUED: RACDEF TYPE=DEFINE, NEWNAME FROM NODE1.RACFU01

 REQUEST OUTPUT:
 IRRR101I Application update request completed successfully
          for class DATASET, profile name MYNEW.PROFILE.
================================================================================
```

Note: NEWNAME is for DATASET or FILE class only (RENAMES).

Sample output for a successful ICHEINTY request:

```
================================================================================
Application update request issued at 15:51:33 on 4/11/98
was processed at NODE2.RACFU01
on 4/11/98 at 15:53:45
Request was propagated by automatic direction from NODE1.RACFU01

 REQUEST ISSUED: ICHEINTY ALTER operation from NODE1.RACFU01

 REQUEST OUTPUT:
```

```
 IRRR101I Application update request completed successfully
          for class $MYCLASS, profile name MYNEW.PROFILE.
================================================================================
```

Sample output for a successful RACROUTE request:

```
================================================================================
Application update request issued at 15:51:33 on 4/13/98
was processed at NODE2.RACFU01
on 4/13/98 at 15:53:45
Request was propagated by automatic direction from NODE1.RACFU01

 REQUEST ISSUED: RACROUTE REQUEST=EXTRACT from NODE1.RACFU01

 REQUEST OUTPUT:
 IRRR101I Application update request completed successfully
          for class $MYCLASS, profile name MYNEW.PROFILE.
================================================================================
```

If a request is unsuccessful or if an abend occurs because of the status of the RACF database on the target, one of three messages is issued:

- IRRR102I if the request type was an ICHEINTY or RACDEF or RACXTRT
- IRRR103I if the request type was a RACROUTE
- IRRR104I if an abend occurred processing a RACROUTE, ICHEINTY, RACDEF, or RACXTRT.

Password synchronization requests initiated by automatic password direction return the following output:

```
================================================================================
Password synchronization request issued at 15:03:58 on 04/18/98 was
processed at NODE2.TSOUSER on 04/18/98 at 15:04:00
Request was propagated by automatic direction from NODE1.TSOUSER

 REQUEST ISSUED: From user TSOUSER at NODE1

 REQUEST OUTPUT:
 IRRC013I Password synchronized successfully for TSOUSER at NODE2 and
 TSOUSER at NODE1.
================================================================================
```

The general output format of error messages is the same.

Interactions among Automatic Direction Functions and Password Synchronization

Each of the automatic direction functions is controlled independently of the other two. Details of these types of interactions follow:

- Between automatic direction of commands and automatic direction of application updates
- Between automatic password direction and automatic direction of application updates
- Among password synchronization, automatic direction of commands, and automatic password direction

Interaction between Automatic Direction of Commands and Automatic Direction of Application Updates

These two automatic directions work independently of each other. If automatic direction of commands is active and automatic direction of application updates is not active, only RACF command updates are propagated to remote nodes. If

automatic direction of application updates is active and automatic direction of commands is not active, only application updates are propagated to remote nodes. If both automatic direction of application updates and automatic direction of commands are active, application updates and commands are propagated to remote nodes.

Interaction between Automatic Password Direction and Automatic Direction of Application Updates

Automatic password direction propagates user password updates. Automatic direction of application updates does *not* propagate user password updates. If updates to non-password-related fields are made with the same ICHEINTY macro execution that updates the password, the propagation of the non-password-related fields is controlled by automatic direction of application updates.

A single ICHEINTY macro `TYPE='USR'` with `ACTIONS=` that specifies both password and non-password user information will result in the propagation of two requests to the target node: one request (to define the user) is propagated by automatic direction of application updates, and the other (to specify password information for the same user) is propagated by automatic password direction. Requests propagated by automatic direction of application updates execute at the target node using the authority of the user ID associated with the application that issued the ICHEINTY to define the user. Requests propagated by automatic password direction execute at the target node using the authority of the user whose password information is to be changed. Because these two requests execute using the authority of different user IDs, they can execute concurrently with unpredictable results.

Unpredictable results may occur with propagation of password and non-password user information through any combination of ICHEINTY macro executions, such as a program executing a single ICHEINTY, or multiple ICHEINTY executions within the same or different programs. For this reason, the recommended methods for defining RACF users are:

1. Execute the ADDUSER command
2. Invoke the `R_admin` callable service from an application program

Automatic password direction may be used to propagate a password update for a user only when that user is defined to RACF on both the source and target nodes.

Interaction among Password Synchronization, Automatic Direction of Commands, and Automatic Password Direction

Password synchronization, automatic direction of commands, and automatic password direction can be active at the same time. These functions interact as follows:

- When a password change is made at logon:
  - Password synchronization sends the password change to users with approved PEER PWSYNC associations, if the user changing the password is authorized to the PWSYNC profile in the RRSFDATA general resource class.
  - If automatic password direction is enabled on the system, and the user who changed the password is authorized to one or more profiles, the password change is automatically directed to the same user IDs on the nodes defined by the RRSFDATA profiles that protect AUTODIRECT.*target-node*.USER.PWSYNC.
- A password changed by the PASSWORD command is not directed by automatic password direction. It is directed by automatic direction of commands based on RRSFDATA profile setup. Also, if the user is authorized to the PWSYNC profile in

the RRSFDATA class, the password changes are sent to users with approved PEER PWSYNC associations with the user whose password has been changed.

- Password synchronization and automatic password direction do not handle password updates initiated by the ADDUSER command. Users who participate in password synchronization must be initially defined to RACF before automatic direction can occur. The ADDUSER command can be directed by automatic direction of commands based on RRSFDATA profile setup.
- The ALTUSER and PASSWORD commands must be automatically directed to maintain the synchronization of user passwords for the same user IDs across RRSF nodes. The automatic direction of commands RRSFDATA profiles that protect AUTODIRECT.*node*.USER.ALTUSER and AUTODIRECT.*node*.USER.PASSWORD control this automatic direction. The RRSFDATA profile that protects automatic password direction (AUTODIRECT.*node*.USER.PWSYNC) is not checked for the automatic direction of the ALTUSER and PASSWORD commands.
- A password change by other methods, such as at logon or by an installation-written application, is not directed by automatic direction of commands. These password changes are sent to users with the same name if automatic password direction is in effect. Also, if the user is authorized to the PWSYNC profile in the RRSFDATA class, the password changes are sent to users with approved PEER PWSYNC associations with the user whose password has been changed.

Using Automatic Direction of Commands

Automatic direction of commands has unique features. These are listed below.

Commands Not Eligible for Automatic Direction of Commands

In general, commands that are ineligible for command direction are also ineligible for automatic direction of commands. Also, commands that do not update the RACF database are ineligible for automatic direction of commands. The following commands are not eligible for automatic direction of commands:

BLKUPD
DISPLAY
LISTDSD
LISTGRP
LISTUSER
RACDCERT
RACLINK
RESTART
RLIST
RVARY
SEARCH
SET
SETROPTS LIST (with no other operands specified)
SIGNOFF
STOP
TARGET

RACF commands can be automatically directed when they are issued in any way except from the RACF parameter library. For example, if they are issued from a TSO session, from a batch job, or even as MVS operator commands, they are still eligible for automatic direction of commands.

How Automatic Direction of Commands Works

Automatic direction of commands:

- Can take place only if SET AUTODIRECT has been used to activate it.
- Does not use or require user ID associations.
- Requires proper authority as defined in the RRSFDATA class.
- Begins after a command has successfully completed with a return code that is less than 8.
- Is relative to where a command runs, as opposed to where it originated.
- Only occurs *from* the node where the command originally runs. Although a command can be automatically directed to multiple nodes, it is not directed or propagated *beyond* those nodes.

Automatic command direction is activated using the SET command and the AUTODIRECT operand to specify output options. See *z/OS SecureWay Security Server RACF Command Language Reference* for more information about the SET AUTODIRECT command.

When automatic command direction is active, and a command runs successfully (with a return code that is less than 8), the command becomes a candidate for automatic direction. Before it is automatically directed to any other nodes, authorization checks are made against profiles in the RRSFDATA class. The authorization checks determine whether the command should be automatically directed, and which nodes it should be directed to.

Automatic Command Direction Authorization Checks

The following example illustrates how automatic direction of commands works:

1. Suppose:
   - NODE1, NODE2, and NODE3 are RRSF nodes that are operative targets of each other.
   - NODE2 and NODE3 have automatic command direction activated between them with the following RRSFDATA class profiles:
     – On NODE2: AUTODIRECT.NODE3.* with UACC(READ)
     – On NODE3: AUTODIRECT.NODE2.* with UACC(READ)
   - CHARLIE2 exists on NODE2 and NODE3, but with no user ID association between nodes.
2. CHARLIE2 on NODE2 issues the following command:

   ```
   ADDUSER PREMA
   ```
3. On NODE2, the ADDUSER PREMA command runs under the authority of CHARLIE2.
4. After the ADDUSER PREMA command runs successfully (under the authority of CHARLIE2 at NODE2), it is automatically directed to NODE3.CHARLIE2.
5. At NODE3, the ADDUSER PREMA command runs under the authority of CHARLIE2.

Notes:

1. The ADDUSER PREMA command is not automatically directed to NODE1.CHARLIE2 because there is no profile protecting the resource AUTODIRECT.NODE1.USER.ADDUSER.
2. The destination of notification and output from the ADDUSER PREMA command that ran on NODE3 is determined by what was specified on SET AUTODIRECT command issued on NODE3.

3. Once the ADDUSER PREMA command runs on NODE3, it is not automatically directed back to NODE2. RACF detects that the command was already automatically directed, and does not further send it to any other nodes.

The AT Option and Automatic Command Direction

The AT option is used to explicitly direct a command to a user ID on a particular RRSF node. Although explicit command direction using the AT option and automatic command direction are two distinct forms of command direction, they can both occur when a command is issued within an RRSF network, as shown in the following examples.

1. Suppose:
   - NODE1, NODE2, and NODE3 are RRSF nodes that are operative targets of each other.
   - NODE2 and NODE3 have automatic command direction activated between them with the following RRSFDATA class profiles:
     - On NODE2: AUTODIRECT.NODE3.* with UACC(READ)
     - On NODE3: AUTODIRECT.NODE2.* with UACC(READ)
   - CHARLIE1 on NODE1 has a peer user ID association with CHARLIE2 on NODE2.
   - CHARLIE2 exists on NODE2 and NODE3, but with no user ID association between nodes.
2. CHARLIE1 on NODE1 issues the following command:

   ```
   ADDUSER PREMA AT(NODE2.CHARLIE2)
   ```
3. Because the AT option was specified, the command is explicitly directed to NODE2.CHARLIE2.
4. At NODE2, the ADDUSER PREMA command runs under the authority of CHARLIE2.
5. After the ADDUSER PREMA command runs successfully (under the authority of CHARLIE2 at NODE2), it is automatically directed to NODE3.CHARLIE2.

Notes:

1. The ADDUSER PREMA command does not run on NODE1.
2. NODE1.CHARLIE1 receives output via the RRSFLIST dataset for the ADDUSER PREMA command that ran on NODE2.
3. The destination of notification and output from the ADDUSER PREMA command that ran on NODE3 is determined by what was specified on SET AUTODIRECT command issued on NODE3.

Summary of Rules for Automatic Direction of Commands

Here is the processing flow of checks that are made to determine whether or not a command should be automatically directed:

1. When a command is issued:
   - If AT is not specified, the command runs on the local node in the user's address space.
   - If AT is specified, the command runs in the RACF subsystem address space of the specified local or remote node.
   - If appropriate, automatic direction of commands occurs from the node where the command executed.
2. The command is not automatically directed if any of these is true:
   a. Automatic direction of commands is inactive.
   b. The command return code is greater than 4.

c. The command has already been automatically directed.

d. The command is ineligible for automatic direction of commands. See “Using Automatic Direction of Commands” on page 378 for more information.

e. The RRSFDATA class is INACTIVE.

f. The RRSFDATA class is ACTIVE and an AUTODIRECT profile covering that command does not exist.

3. For each remote target node, the following occurs:

   a. If the command issuer does not have at least READ authorization to RRSFDATA profile AUTODIRECT.*target-node.classname.command-name*, the command is not automatically directed to this node

   b. If the command has passed all checks so far, it is sent to execute on the remote node under the authority of the same-named user ID on the remote node. The user ID is the user ID under which the command executed, which is not necessarily the command issuer (if the command was directed and then automatically directed, for example).

      For example, a command issued by and executed on LAURIE at NODE1 is automatically directed to LAURIE at NODE2. A command issued by LAURIE specifying AT(NODE2.ANDREW) is automatically directed to ANDREW at NODE1. No authorization check with the AUTODIRECT profiles is made on the receiving node.

Using Automatic Direction of Application Updates

See “Controlling Automatic Direction of Passwords” on page 286 for information about the profiles needed for automatic direction of application updates.

Summary of Rules for Automatic Direction of Application Updates

1. When automatic direction of application updates is active, RACF automatically directs selected application updates made by the following commands and macros to selected remote nodes:
   - ICHEINTY ADD, ALTER, DELETE, DELETEA, and RENAME requests
   - The RACDCERT command
   - RACDEF
   - RACROUTE REQUEST=DEFINE
   - RACROUTE REQUEST=EXTRACT,TYPE=REPLACE
   - RACXTRT specifying TYPE=REPLACE
2. If profiles on two or more RRSF nodes are already synchronized, you can use automatic direction of application updates to keep the profiles synchronized with respect to application updates.
3. RACF directs an application update only after the update has successfully completed on the node where the application is executing.
4. Not all RACROUTE REQUEST=DEFINE and RACDEF requests update the RACF database, and RACF does not automatically direct requests that do not update the database. RACROUTE REQUEST=DEFINE and RACDEF are not automatically directed if:
   - ENVIR=VERIFY is specified
   - RACFIND=NO is specified and DSTYPE=T is not specified
5. RACROUTE REQUEST=VERIFY and RACINIT update the RACF database by issuing ICHEINTY macros. RACF automatically directs the following ICHEINTY requests made by RACROUTE REQUEST=VERIFY and RACINIT:

- The ICHEINTY setting the revoke flag in the user profile when a user is being revoked due to inactivity or password attempts that are not valid
- The ICHEINTY that increments the revoke count when a user enters a password that is not valid
- The ICHEINTY that resets the revoke count to 0 when a user enters a valid password, if the revoke count for the user was nonzero before the update was made

6. Automatic direction of the ICHEINTY that RACROUTE REQUEST=VERIFY issues to change the password in the user's profile is controlled by automatic password direction, and not by automatic direction of application updates.
7. When a RACROUTE REQUEST=DEFINE, or a RACDEF, issues an ICHEINTY, RACF does not direct the ICHEINTY separately.
8. Automatic command direction determines whether a RACF command is directed. If a command issues a RACROUTE or ICHEINTY, that RACROUTE or ICHEINTY is not directed by automatic direction of application updates.
9. Use the AUTOAPPL and NOAUTOAPPL options on the RACF SET command to activate and deactivate automatic direction of application updates. Use the OUTPUT and NOTIFY values of SET AUTOAPPL to specify which users will be notified of results and receive output from automatically directed application updates.
10. Profiles in the RRSFDATA class control which application updates are automatically directed to which nodes.

Considerations for the DATASET Class

Because discrete DATASET profiles are closely tied to the data set they protect and DATASET profiles for tape data sets are closely tied to profiles in the TAPEVOL class, the following are important:

- If a discrete profile is created without turning on the RACF indicator bit for the dataset in the VTOC or catalog entry, the profile is not found during authority checking.
- If a discrete data set profile is renamed and the data set itself is not renamed, the discrete profile no longer is used for data set protection.
- If a discrete data set profile is deleted and the data set itself is not deleted, its protection is likely to be changed to an existing generic profile.
- Because RACROUTE REQUEST=DEFINE and RACDEF only manipulate the RACF profiles and do not modify the data set itself or its RACF indicator bit, it is not desirable to propagate these changes to a remote system.

Because these changes are controlled by the AUTODASD and AUTOTAPE profiles, it is not necessary to turn off propagation of

`AUTODIRECT.`*`node`*`.DATASET.APPL`

unless your own applications require this.

Neither application update requests nor application update output can be sent to a node running RACF 2.2. The output is lost, and the attempts degrade performance. This includes the output to the RRSFLIST dataset and the TSO sends (notify). Message IRRR119I indicates an incorrect attempt to send information to a system unable to receive it.

RRSF Considerations for Digital Certificates

Updates can be made to digital certificate information in the RACF database by the RACDCERT, RDEFINE, RALTER and RDELETE commands, and by applications that invoke the `R_datalib` and `initACEE` callable services. These updates can affect

profiles in DIGTCERT, DIGTCRIT, DIGTNMAP, DIGTRING and USER classes. Since these updates are eligible for automatic direction, you must ensure consistent propagation across these classes.

Propagation of Command and Application Updates: To ensure RACF database updates are propagated in a consistent manner across the DIGTCERT, DIGTCRIT, DIGTNMAP, DIGTRING and USER classes, you should create a single RRSFDATA profile called AUTODIRECT.*target-node*.DIGT*, or you should ensure that the access lists for the RRSFDATA resources shown in Table 27 are kept identical:

Table 27. RRSFDATA Resources Used to Control Propagation of Digital Information

Type of automatic direction	RRSFDATA resource
Automatic direction of application updates	AUTODIRECT.*target-node*.DIGTCERT.APPL
	AUTODIRECT.*target-node*.DIGTNMAP.APPL
	AUTODIRECT.*target-node*.DIGTRING.APPL
Automatic direction of commands	AUTODIRECT.*target-node*.DIGTCRIT.*command-name*
Note: The best way to ensure that RACF database updates are propagated consistently is to define a single profile called AUTODIRECT.*target-node*.DIGT* to control all the RRSFDATA resources shown above.	

To facilitate consistency, when updates are made in the USER class that pertain to digital certificates, the RRSFDATA resource AUTODIRECT.*target-node*.DIGTCERT.APPL is used to determine the authority to propagate. This resource should be protected by the AUTODIRECT.*target-node*.DIGT* resource profile. Note that the AUTODIRECT.*target-node*.USER.APPL resource is not checked to determine authority to propagate USER-class updates that are made as a result of processing digital certificates.

Suppression of Private Key Information Propagation

When automatic direction of application updates is enabled, RACF database changes affecting certificate information can be propagated to other systems. However, the private key information about certificates contained in the following fields of general resource profiles in DIGTCERT class is not propagated:

CERTPRVK Private key

CERTPRVS Private key size

CERTPRVT Private key type

RACF does not store the private key associated with a certificate. Instead, RACF stores the Integrated Cryptographic Service Facility (ICSF) key token. The private key itself is stored in the ICSF public key data set (PKDS) and is encrypted under the master key of the system. Since the automatic direction of application updates propagates information to remote systems that are likely to have different public key data sets and different master keys, private key information from the originating node is unusable.

Using Automatic Password Direction

When PEER PWSYNC changes a password, the change is not propagated to other nodes by automatic password direction. However, when automatic password direction changes a password, that change results in PEER PWSYNC changing the password on all of the user IDs that have the proper association and profile access. Automatic password direction and PWSYNC work best together when peer

associations exist only with other user IDs on the local system and the same set of user IDs are associated with each other on each other system. For example, NODE1.JOE1 and NODE1.JOE2 have peer PWSYNC and NODE2.JOE1 and NODE2.JOE2 have peer PWSYNC with each other (but not with Joe's user IDs on node1). When Joe changes his password on NODE1.JOE1, peer PWSYNC carries it on to JOE2. This keeps both network traffic and complexity of associations to a minimum, while keeping all of Joe's passwords the same.

If you are using automatic password direction between same named users on your system and another RRSF node, do not establish PEER PWSYNC user associations between the same user IDs across RRSF systems that use automatic password direction. Doing so would result in duplication of password synchronization requests.

Relationship to User ID Associations

No user ID associations are required for automatic password direction. If user ID associations are present, passwords are synchronized for the users with approved PEER PWSYNC associations to the user who initiated the password change.

PWSYNC associations can be used in environments with automatic password direction. In this environment, the passwords of users who have PEER PWSYNC associations with the originator of the password change are synchronized with the originator's password. Automatic password direction only synchronizes passwords between the same user IDs on multiple RRSF nodes.

RRSF Considerations for Network Authentication Service

If your installation has implemented automatic direction and you wish to define multiple realms, you should review your current RRSF implementation in view of these important considerations:

1. The KERB segment of the RACF user profile defines a user as a local principal. If KERB segment information is directed to a remote RRSF node, users will be defined as local principals on all Network Authentication Service servers that share that RACF database.
2. RACF does not distinguish between user passwords and passwords assigned to local principals for key generation. If user passwords are synchronized with a remote RRSF node, keys will be generated for those users on the remote node and they will be recognized as local principals by all Network Authentication Service servers that share that RACF database.
3. REALM class profiles define information about local and foreign realms. If these profiles are propagated to a remote RRSF node, all Network Authentication Service servers that share that RACF database will have duplicate local and foreign realm definitions.
4. KERBLINK class profiles define the mapping of foreign principals to local RACF user IDs. If these profiles are propagated to a remote RRSF node, all Network Authentication Service servers that share that RACF database will attempt to map those foreign principals to the same RACF user IDs.

See "Chapter 20. RACF and SecureWay Network Authentication Service" on page 555 for more information.

Synchronizing Database Profiles

You can use automatic direction to maintain synchronization of RACF database profiles that are already synchronized, but you must synchronize the profiles before you activate RRSF functions. You can do this synchronization manually, but it can be a time-consuming process. You can also run IRRDBU00 against the databases you want to synchronize, and use a program or REXX EXEC to compare the IRRDBU00 output for the databases and generate the commands needed to synchronize them. IBM provides a sample REXX EXEC, DBSYNC, to help you do this. IBM does not support the DBSYNC EXEC. For information on how to get this tool and others from the RACF home page or via anonymous FTP, see “Internet Sources” on page xxv.

Chapter 11. Controlling Access to DB2 Objects

This chapter provides details about using RACF to provide security for DB2 objects.

Additional Reading

Before you read this chapter, you should have a basic understanding of DB2, DB2 security concepts, and configuring the RACF/DB2 external security module. For information about:

- DB2 and DB2 security concepts, see the appropriate DB2 publication for your installation—for example:
  - *DB2 for MVS/ESA Version 4 Administration Guide,* SC26-3265
  - *DB2 for OS/390 Version 5 Administration Guide,* SC26-8957
  - *DB2 Universal Database for OS/390 Version 6 Administration Guide,* SC26-9003
  - *DB2 Universal Database for OS/390 and z/OS Version 7 Administration Guide,* SC26-9931
- Configuring the RACF/DB2 external security module, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

RACF Support for DB2 Authorization

RACF support for DB2 authorization includes the following:

- The RACF/DB2 external security module, which is provided as a sample assembler language routine and is shipped as SYS1.SAMPLIB(IRR@XACS). This sample program is intended to be invoked as the DB2 access control authorization exit and is a replacement for the default routine shipped with DB2. See *z/OS SecureWay Security Server RACF System Programmer's Guide* for more information on installing the RACF/DB2 external security module.
- A set of 29 general resource classes provided in the supplied class descriptor table, ICHRRCDX, to be used by the RACF/DB2 external security module (when customization options contain default values).

See “Supplied Resource Classes for z/OS and OS/390 Systems” on page 565 for more information.

- A set of 29 RACF router table entries provided in the supplied RACF router table, ICHRFR0X.

  See *z/OS SecureWay Security Server RACF Macros and Interfaces* for more information.

Notes:

1. This chapter refers to various fields in two parameter lists that are passed to the RACF/DB2 external security module: EXPL (DSNDEXPL) and XAPL (DSNDXAPL). The EXPL is DB2’s general exit parameter list and XAPL is the parameter list specific to the authorization exit. For more information, see the *DB2 Administration Guide*.
2. For DB2 data sharing, the DB2 group attachment name is used in place of the DB2 subsystem name.

Configuring the RACF/DB2 External Security Module

The steps for configuring the RACF/DB2 external security module include the following:

1. With the system programmer and the DB2 database administrator, choose the customization options.

   See the *z/OS SecureWay Security Server RACF System Programmer’s Guide* for details.
2. Optionally, the system programmer sets the customization options.

   See the *z/OS SecureWay Security Server RACF System Programmer’s Guide* for details.
3. Optionally, the system programmer defines general resource classes.

   See the *z/OS SecureWay Security Server RACF System Programmer’s Guide* for details.
4. Choose which DB2 objects will be protected by the RACF/DB2 external security module.

   See “Migrating to the RACF/DB2 External Security Module” on page 390 for details.
5. Define profiles for the DB2 objects that will be protected by the RACF/DB2 external security module.

   See “Protecting DB2 Objects” on page 391 for details.
6. Defines profiles for the DB2 administrative authorities.

   See “DB2 Administrative Authorities” on page 396 for details.
7. Activate the general resource classes for the DB2 objects you have chosen to protect.

   See “Initialization” on page 405 for details.
8. The system programmer assembles and link-edits the RACF/DB2 external security module.

   See the *z/OS SecureWay Security Server RACF System Programmer’s Guide* for details.
9. The DB2 database administrator restarts the DB2 subsystem.

   See the *DB2 Administration Guide* for details.

Migrating to the RACF/DB2 External Security Module

When migrating from DB2 internal security to the RACF/DB2 external security module, it is not necessary to migrate protection of all DB2 objects at once. It is possible to begin using the RACF/DB2 external security module before defining profiles to protect all DB2 object types. When the RACF/DB2 external security module determines that there is no profile to protect a particular DB2 object or that the class corresponding to a particular DB2 object type is not active, it will defer to DB2 for authority checking.

For example, suppose only the set of RACF profiles to protect DB2 tables has been defined and the classes for all other object types have not been made active. In this case, the RACF/DB2 external security module performs profile checking for DB2 tables and defers to DB2 for authority checking of other object types, such as plans, packages, and databases.

It is also recommended that profiles for the DB2 administrative authorities be defined before the RACF/DB2 external security module is activated. When the RACF/DB2 external security module determines that there is a profile protecting a DB2 object and the requesting user has not been permitted access to the object, it will indicate to DB2 that the requesting user does not have authority to the object (regardless of whether the relevant DB2 administrative authority profiles exist). When this occurs, DB2 does not have the opportunity to allow access based on the requesting user having an administrative authority in DB2.

RACF Profile Checking

Each DB2 command, utility, and Structure Query Language (SQL) statement is associated with a set of sufficient privileges, authorities, or both.

Authority checking performed by the RACF/DB2 external security module simulates DB2 authority checking:

- DB2 object types map to object class names in RACF
- DB2 privileges map to object profile names in RACF
- DB2 authorities map to administrative authority class and profile names in RACF
- DB2 security rules map to profile checking in RACF

The RACF/DB2 external security module checks the RACF profiles corresponding to that set of privileges and authorities.

See “Special Considerations” on page 399 and “Appendix D. RACF/DB2 External Security Module: Authorization Checking” on page 595 for more information.

Matching Schema Names

Certain privileges associated with schema objects (such as user-defined functions, user-defined distinct types, and stored procedures), can be granted if the user identity *matches* the schema name. The schema name is a short SQL identifier used as a qualifier in the name of schema objects and creates a logical grouping of these objects. It is often, but not always, an DB2 authorization ID. For applicable privileges, RACF/DB2 external security module will look for a match on schema name before checking RACF profiles.

For authorization checking of the CREATEIN schema privilege, the RACF/DB2 external security module first checks to see if the user identity in field XAPLUCHK

matches the schema name in XAPLOBJN. If those two fields are equal, the RACF/DB2 external security module allows the access. For all other schema privileges, the RACF/DB2 external security module first checks to see if the user identity in XAPLUCHK matches the schema name in XAPLOWNQ. If those two fields are equal, the RACF/DB2 external security module allows the access. In each case, when the RACF/DB2 external security module allows access, it returns a return code 0 in EXPLRC1 and reason code 14 in EXPLRC2, and no further checking occurs. If the RACF/DB2 external security module does not allow the access, profile checking occurs. See “Appendix D. RACF/DB2 External Security Module: Authorization Checking” on page 595 for details.

Protecting DB2 Objects

The resources that apply to a particular invocation of the RACF/DB2 external security module depend on the input object type (XAPLTYPE) and the privilege being checked (XAPLPRIV). The object types and their associated privileges are shown in “Appendix D. RACF/DB2 External Security Module: Authorization Checking” on page 595. See *DB2 Administration Guide* to find the numeric XAPLPRIV values (used by the RACF/DB2 external security module) that correspond to the privilege names.

The general resource class and profile names used by the RACF/DB2 external security module to protect DB2 objects depend on the options specified by these assembler SET symbols:

SET Symbol	Default Value	Description
&CLASSOPT	2	Specifies the classification model to be used
&CLASSNMT	DSN	Specifies the class name root to be used
&CHAROPT	1	Specifies the class name suffix to be used

The &CLASSOPT, &CLASSNMT, and &CHAROPT options specify the format of the class names and resource profile names used by the RACF/DB2 external security module. These options are global for each DB2 subsystem, and must be the same for all classes. Each instance of the RACF/DB2 external security module can only be set up to process one classification model or the other, but not both. See *z/OS SecureWay Security Server RACF System Programmer’s Guide* for more information.

If your installation is using the default values for these options, you can use the classes in the supplied class descriptor table (ICHRRCDX). Additional classes do not need to be defined.

Security administrators need to define the correct profiles according to which options are being set in the RACF/DB2 external security module.

Class Names

There is one general resource class associated with each DB2 object type. There are 13 DB2 object types. In the supplied class descriptor table (ICHRRCDX), two classes are defined for each object type, so that each object type has an associated member class and an associated grouping class. Table 28 on page 393 lists the supported DB2 objects and class abbreviations.

Installations defining their own classes can also define two classes for each object type if member and grouping classes are desired. If only one class is defined for each object type, the class name must begin with M (*not* G).

The actual format of the class names of DB2 objects depends on the classification model being used.

Note: An additional class can be defined to support DB2 administrative authorities. See section “DB2 Administrative Authorities” on page 396.

System programmers can alter the &CLASSOPT field of the modifiable assembler source statement in the RACF/DB2 external security module to select the classification model their installation will use:

Classification Model (&CLASSOPT Value)	Scope
1	Single-subsystem scope
2	Multiple-subsystem scope **Note:** *This is the default.*

Once a classification model is selected, the system programmer can use the &CHAROPT and &CLASSNMT SET symbols to alter the default naming conventions for the general resource classes. The naming conventions for the resource profile names are defined in “Resource Names” on page 394.

Classification Model 1: Single-Subsystem Scope

When this model is selected, the RACF/DB2 external security module places the DB2 subsystem name in the class name. Class names have this format:

ayyyyxxz

where:

a Is M for member class or G for grouping class

yyyy Is the DB2 subsystem name or, if data sharing, the DB2 group attachment name (from XAPLGPAT)

xx Is the type of DB2 object (see Table 28 on page 393 for valid values)

z Is the &CHAROPT value (the default is 1)

The resource profile names do not contain the subsystem name.

Installations using single-subsystem scope cannot use the classes provided in the supplied class descriptor table (ICHRRCDX). They must define their own classes in the installation-defined class descriptor table (ICHRRCDE), unless they are using the default DB2 subsystem name of DSN and have altered the &CHAROPT variable in the RACF/DB2 external security module to be a blank character (' '). In addition, the installation must define a separate set of classes for each subsystem that will be using the RACF/DB2 external security module.

Classification Model 2: Multiple-Subsystem Scope (Default)

When this model is selected, the RACF/DB2 external security module places the DB2 subsystem name in the resource name. Class names have this format:

abbbbxxz

where:

a Is M for member class or G for grouping class

bbbb Is the &CLASSNMT value (the default value is DSN)

xx Is the type of DB2 object (see Table 28 for valid values)

z Is the &CHAROPT value (ignored if &CLASSNMT='DSN')

The resource names are prefixed with the DB2 subsystem name or, if data sharing, DB2 group attachment name.

Installations using multiple-subsystem scope and the default &CLASSNMT value (DSN) can use the classes provided in the supplied class descriptor table (ICHRRCDX). Any subsystem sharing the RACF/DB2 external security module can use the same set of classes. A separate set of classes does not need to be defined for each subsystem.

If &CLASSNMT is set to a value other than the default (DSN), classes must be defined in the installation-defined class descriptor table (ICHRRCDE). If only one class is defined for each object type, the class name must begin with M (*not* G).

DB2 Object Types

Each authorization request has an associated object type. DB2 provides the object type as a 1-character abbreviation in the XAPLTYPE field. This abbreviation is used by the RACF/DB2 external security module in conjunction with the privilege code (see “Privilege Names” on page 395) to determine which checking to perform.

A non-valid XAPLTYPE or XAPLPRIV passed to the RACF/DB2 external security module during authorization checking will cause the RACF/DB2 external security module to return a return code of 4 (“RACF access not determined; perform DB2 access checking”).

Table 28 lists the DB2 objects, the DB2 abbreviations used in the XAPL, and the abbreviations used in the RACF general resource grouping and member class names (GDSN*xx* and MDSN*xx*):

Table 28. DB2 Object Abbreviations

DB2 Object	DB2 Object Abbreviation	RACF Class Abbreviation
Buffer pool	B	BP
Collection	C	CL
Database	D	DB
Java archive (JAR)	J	JR
Package	K	PK
Plan	P	PN
Schema	M	SC
Storage group	S	SG
Stored procedure	O	SP
System	U	SM
Table, index, view	T	TB
Table space	R	TS
User-defined distinct type	E	UT
User-defined function	F	UF

Resource Names

The format of the resource names built by the RACF/DB2 external security module depends on the classification model being used.

For single-subsystem scope the general format is:

[*object-name*.]*privilege-name*

For multiple-subsystem scope the general format is:

DB2-subsystem.[*object-name*.]*privilege-name*

or, if data sharing:

DB2-group-attachment-name.[*object-name*.]*privilege-name*

For multiple-subsystem scope, the DB2 subsystem name is obtained from XAPLGPAT.

The object name used depends on the object type and privilege. The valid privilege qualifiers are defined in section “Privilege Names” on page 395.

Object Names

The RACF/DB2 external security module constructs the DB2 object resource name using information passed in various fields (XAPLOBJN, XAPLOWNQ, XAPLREL1, and XAPLREL2). The content of these fields depends on the input object type, XAPLTYPE.

Table 29 defines the DB2 object name qualifiers used in RACF profiles:

Table 29. DB2 Object Name Qualifiers for RACF Profiles

DB2 Object	Object Name Qualifiers
Buffer pool	*bufferpool-name*
Collection	*collection-ID*
Database	*database-name*
Java archive (JAR)	*schema-name.JAR-name*
Package	*collection-ID.package-ID* *collection-ID* *owner*
Plan	*plan-name* *owner*
Schema	*schema-name* *schema-name.function-name* *schema-name.procedure-name* *schema-name.type-name*
Storage group	*storage-groupname*
Stored procedure	*schema-name.procedure-name*
System	*owner* (BINDAGENT only)
Table, index, view	*table-owner.table-name* *table-owner.table-name.column-name*
Table space	*database-name.table-space-name*
User-defined distinct type	*schema-name.type-name*
User-defined function	*schema-name.function-name*

Note: The format of the DB2 object names is defined by DB2. For more information, see *DB2 SQL Reference*.

Privilege Names

The RACF/DB2 external security module constructs an object resource name using the DB2 privilege as the lowest-level qualifier (RACF profile-name suffix) in the resource name. Each explicit privilege used as a low-level qualifier corresponds to one of the explicit privilege names that DB2 uses for a particular object. For more information about explicit DB2 privileges, see the *DB2 Administration Guide*.

Notes:

1. The RACF/DB2 external security module does not support GRANT ALL for any object type.
2. The RACF/DB2 external security module can handle *all* privileges in the following lists. At this time, however, an input ACEE is not available to the RACF/DB2 external security module for those privileges followed by an asterisk (*****). For more information on the conditions for which there is no input ACEE, see the *DB2 Administration Guide*.

Table Privileges: The valid table privileges used as suffixes are:
ALTER
DELETE
INDEX
INSERT
REFERENCES
SELECT
TRIGGER
UPDATE

Plan Privileges: The valid plan privileges used as suffixes are:
BIND
EXECUTE

Package Privileges: The valid package privileges used as suffixes are:
BIND
COPY
EXECUTE

Collection Privileges: The valid collection privilege used as a suffix is:
CREATEIN

Database Privileges: The valid database privileges used as suffixes are:
CREATETAB
CREATETS
DISPLAYDB*
DROP
IMAGCOPY
LOAD
RECOVERDB
REORG
REPAIR
STARTDB*
STATS
STOPDB*

System Privileges: The valid database privileges used as suffixes are:

ARCHIVE*
BINDADD
BINDAGENT
BSDS*
CREATEALIAS
CREATEDBA
CREATEDBC
CREATETMTAB
CREATESG
DISPLAY*
MONITOR1
MONITOR2
RECOVER*
STOPALL*
STOSPACE
TRACE*

Buffer Pool, Storage Group, and Table Space Privileges: The valid privilege used as a suffix is:

USE

Java Archive (JAR) and User-Defined Distinct Type Privileges: The valid privilege used as a suffix is:

USAGE

User-Defined Function and Stored Procedure Privileges: The valid privilege used as a suffix is:

DISPLAY*
EXECUTE

Schema Privileges: The valid privilege used as a suffix is:

ALTERIN
CREATEIN
DROPIN

DB2 Administrative Authorities

The RACF/DB2 external security module supports the DB2 concept of administrative authorities. DB2 administrative authorities often include privileges that are not explicit, have no name, and cannot be specifically granted. For example, the ability to terminate any utility job is included in the SYSOPR authority.

If users were not permitted access to the object through the object resource profile, subsequent checks are made to determine if the user has been permitted access to system resources via the DSNADM class profiles. The *DB2 Administration Guide* documents the set of privileges each DB2 administrative authority provides. The administrative authorities that apply to a particular invocation of the RACF/DB2 external security module, depend on the input object type (XAPLTYPE) and the privilege being checked (XAPLPRIV) and are defined in “Appendix D. RACF/DB2 External Security Module: Authorization Checking” on page 595.

Like the names used to protect DB2 objects, the general resource class and profile names used to implement DB2 administrative authorities depend on the options specified with the assembler SET symbols.

Class Names

A RACF class, called the DB2 administrative authority class or DSNADM (the default class name), allows RACF security administrators to create profiles that are suffixed with specific DB2 administrative authorities, to allow users to access certain resources for specified DB2 subsystems or groups. The format is dependent on the scope (&CLASSOPT) specified.

Classification Model 1: Single-Subsystem Scope

When this model is selected, the RACF/DB2 external security module places the DB2 subsystem name in the class name. Class names have this format:

yyyyADMz

where:

yyyy	Is the DB2 subsystem name or, if data sharing, the DB2 group attachment name (from XAPLGPAT)
ADM	Is the class abbreviation for administrative authority
z	Is the &CHAROPT value (the default value is 1)

The resource profile names do not contain the subsystem name.

Installations using single-subsystem scope cannot use the DB2 administrative authority class provided in the supplied class descriptor table (ICHRRCDX) and must define their own class in the installation-defined class descriptor table (ICHRRCDE), unless they are using the default DB2 subsystem name DSN and have altered the &CHAROPT variable in the RACF/DB2 external security module to be a blank (' '). In addition, the installation must define a separate class for each subsystem that will use the RACF/DB2 external security module.

Classification Model 2: Multiple-Subsystem Scope (Default)

When this model is selected, the RACF/DB2 external security module places the DB2 subsystem name in the resource name. Class names have this format:

yyyyADMz

where:

yyyy	Is the &CLASSNMT value (the default value is DSN)
ADM	Is the class abbreviation for administrative authority
z	Is the &CHAROPT value, which is ignored if &CLASSNMT is set to DSN

The resource names are prefixed with the DB2 subsystem name or, if data sharing, the DB2 group attachment name.

Installations using multiple-subsystem scope and the default &CLASSNMT value (DSN) can use the DB2 administrative authority class provided in the supplied class descriptor table (ICHRRCDX). Any subsystem sharing the RACF/DB2 external security module can use the same class. A separate class does not need to be defined for each subsystem.

If &CLASSNMT is set to a value other than DSN, a class must be defined in the installation-defined class descriptor table (ICHRRCDE).

Resource Names

The format of the resource names built by the RACF/DB2 external security module depends on the classification model being used.

For single-subsystem scope, the format for the DB2 administrative authority resources is:

[*object-name*.]*authority-name*

For multiple-subsystem scope the general format is:

DB2-subsystem.[*object-name*.]*authority-name*

or, if data sharing,

DB2-group-attachment-name.[*object-name*.]*authority-name*

For multiple-subsystem scope, the DB2 subsystem name is obtained from XAPLGPAT. The object name used depends on the DB2 administrative authority.

DB2 Administrative Authorities and Object Names

The RACF/DB2 external security module will construct the DB2 object resource name using information passed in various fields (XAPLOBJN, XAPLOWNQ, XAPLREL1 or XAPLREL2). The content of these fields depends on the input object type, XAPLTYPE.

Table 30 lists the DB2 administrative authorities and the associated RACF object qualifiers:

Table 30. DB2 Administrative Authorities

Administrative Authority	RACF Object Qualifiers
DBADM	*database-name*
DBCTRL	*database-name*
DBMAINT	*database-name*
PACKADM	*collection-ID*
SYSADM	none
SYSCTRL	none
SYSOPR	none

Note: The format of the DB2 object names is defined by DB2. For more information, see *DB2 SQL Reference*.

Installation-Defined Classes

When the &CLASSOPT or &CLASSNMT assembler SET symbols are changed from their default values, classes must be defined in the installation class descriptor table (CDT). An installation must define:

- One class for each of the 13 DB2 object types they want to protect with the RACF/DB2 external security module.

  It is not necessary to define classes for DB2 objects that will not be protected by the RACF/DB2 external security module.
- The DB2 administrative authority class.

Optionally, the installation can define two classes for each object type and set them up as associated member and grouping classes. The formats for these class names are defined in sections “Classification Model 1: Single-Subsystem Scope” on page 392 and “Classification Model 2: Multiple-Subsystem Scope (Default)” on page 392.

When using single-subsystem scope, class names are created dynamically by concatenating the DB2 subsystem name or, if data sharing, the DB2 group attachment name, with the object type. As a result, multiple DB2 subsystems can use the same copy of the RACF/DB2 external security module.

When using multiple-subsystem scope, class names are created dynamically by concatenating the &CLASSNMT value with the object type. As a result, DB2 subsystems with the same &CLASSNMT value can use the same copy of the RACF/DB2 external security module.

Note: If you want to use installation-defined classes, you must use installation-defined classes with all objects for the same copy of the RACF/DB2 external security module. You cannot mix supplied classes and installation-defined classes for the same copy of the RACF/DB2 external security module. You must use different versions of the RACF/DB2 external security module.

Installation-defined classes should have the same class descriptor table (CDT) attributes as the supplied DB2 classes that they correspond to, except for the POSIT value.

DB2 Data Sharing

The RACF/DB2 external security module can be used with DB2 data sharing. When DB2 has been configured for data sharing, it will pass the RACF/DB2 external security module the name of the DB2 data sharing group name in place of the DB2 subsystem name. As a result, class names and profile names must be defined with the DB2 data sharing group name in place of the DB2 subsystem name. To use the RACF/DB2 external security module in this environment, all systems in the DB2 data sharing group must share the same RACF database.

For more information on DB2 data sharing, see *DB2 Data Sharing: Planning and Administration*.

Special Considerations

In certain instances, the RACF authorization checking done by the RACF/DB2 external security module is different from the authorization checking done by DB2. These instances are described in this section.

PUBLIC*

The DB2 user ID PUBLIC* cannot be supported directly in the RACF/DB2 external security module. However, the security administrator can define a resource profile using a generic character in multiple-subsystem scope in place of the DB2 subsystem name, with the appropriate UACC level for the object. This profile would then allow all users from all subsystems to access the resource as desired.

Implicit Privileges of Ownership

When the user is the owner of a DB2 object, the user may have some implicit privileges, but not all privileges associated with the object. The RACF/DB2 external security module supports certain implicit privileges of ownership for the following DB2 objects and associated privileges.

Table 31. DB2 Objects and Privileges associated with Implicit Ownership

DB2 Object	Owner Field	Implicit Privileges
Java archive (JAR)	XAPLREL1	USAGE
Package	XAPLREL1	BINDAUT and COPYAUT
Plan	XAPLOWNQ	BINDAUT
Schema	XAPLREL1	ALTERIN, COMMENT ON, and DROPIN
Stored procedure*	XAPLREL1	DISPLAY, START, and STOP
Table	XAPLOWNQ	All privileges except CRTSYAUT, DRPSYAUT, CRTVUAUT, and CNVRTAUT
User-defined distinct type	XAPLREL1	USAGE
User-defined function*	XAPLREL1	DISPLAY, START, and STOP
Note: An input ACEE is not available to the RACF/DB2 external security module for the privileges associated with those DB2 objects followed by an asterisk (*****). See the *DB2 Administration Guide* for more information on the conditions for which there is no input ACEE.		

To check authorization for the privileges associated with implicit ownership, the RACF/DB2 external security module first checks to see if XAPLUCHK or XAPLUPRM matches the value passed in the owner field. (See Table 31 for the name of the field that DB2 uses to pass owner information for each object type.) If either of these two fields are equal, the RACF/DB2 external security module authorizes access and returns a return code 0 in EXPLRC1 and reason code 13 in EXPLRC2. If this check fails, a check is made to see if XAPLUCHK equals the owner field (“does the current SQL ID equal the owner of the object?”). If these two fields are equal, access is allowed. If this check fails, profile checking will occur.

DROP and ALTER INDEX Privileges

When using DB2 Version 5 or DB2 Version 6, the database name is not available when the RACF/DB2 external security module is invoked to check a user’s authority to drop or alter an index. As a result, the RACF/DB2 external security module cannot check a user’s DBADM authority for the database that contains the index. To be able to drop or alter an index, a user must either own the index *or* have one of the following:

1. SYSCTRL authority
2. SYSADM authority
3. Access to one of the following resources:
   a. For multiple-subsystem scope, in the DSNADM class:
      DB2-subsystem-name.DBADM
   b. For single-subsystem scope, in the *xxx*ADM class:
      DBADM

      where *xxx* is the class name root (&CLASSNMT value).

For the exact class and resource names, see “Appendix D. RACF/DB2 External Security Module: Authorization Checking” on page 595.

When using DB2 Version 7, the database name is available for DBADM authorization checking. Therefore, DBADM authority to the database is sufficient for a user to be able to drop or alter an index.

CREATETMTAB Privilege

In DB2, the DBMAINT, DBCTRL, and DBADM administrative authorities are sufficient for the CREATETMTAB privilege. However, with the RACF/DB2 external security module, a user must have *one* of the following:

1. The CREATETMTAB privilege
2. The CREATETAB privilege
3. SYSCTRL authority
4. SYSADM authority

For the exact class and resource names, see “Appendix D. RACF/DB2 External Security Module: Authorization Checking” on page 595.

CREATE VIEW Privilege

With DB2 Version 5 or DB2 Version 6, DB2 does not use DBADM authority for a database to authorize a user to create a view. Beginning with DB2 Version 7, if the installation option DBADM CREATE VIEW on panel DSNTIPP (ZPARM DBACRVW) is set to YES during DB2 installation, users with DBADM privilege for a database may be allowed to create views for others. See *DB2 Administration Guide* for information about other privileges required to create a view.

When a view is based on tables or a combination of tables and views from more than one database, the view creator must have DBADM for at least one database that contains a table referenced in the view. DB2 Version 7 passes the names of databases referenced in each CREATE VIEW to the RACF/DB2 external security module using a database list pointed to by XAPLREL2.

If SYSCTRL or SYSADM authorization checking does not allow the CREATE VIEW privilege, and the XAPLCRVW field indicates that DBACRVW is enabled, the RACF/DB2 external security module checks the user’s DBADM authorization for each database in the list. The result of each DBADM check is placed in the XAPLDBDA field associated with each database. See *z/OS SecureWay Security Server RACF Diagnosis Guide* for information about capturing the results from the RACF/DB2 external security module.

″Any Table″ Privilege

In DB2, the UPDATE or REFERENCES privilege for a specific column is sufficient to allow the “any table” privilege. When the RACF/DB2 external security module is invoked for the “any table” privilege, having the UPDATE privilege or the REFERENCES privilege is not sufficient to provide the user with the “any table” privilege.

″Any Schema″ Privilege

RACF does not perform authorization checks looking for ″all privileges on all schemas″ as DB2 does for the CREATEIN, ALTERIN, DROPIN, and COMMENT ON privileges on schemas; nor does RACF look for ″all privileges on all stored procedures″ as DB2 does for the EXECUTE privilege for stored procedures. Note that RACF generic profiles can be used to define protection for sets of similarly

named schemas and stored procedures. RACF variables and RACF grouping profiles may be used for the protection attributed of schemas and stored procedures that are not similarly named.

UPDATE and REFERENCES Authorization on DB2 Table Columns

The RACF/DB2 external security module handles UPDATE and REFERENCES authorizations associated with columns by first checking for access to the entire table (example: *table*.UPDATE) and if not permitted, then to each individual column (example: *table.column*.UPDATE).

When performing an authorization check on a column privilege, the RACF/DB2 external security module informs DB2 if access is allowed because it is granted on the whole table or through an individual column. In DB2, this check is performed using fields UPDATECOLS and REFCOLS. The RACF/DB2 external security module returns a value to DB2 in output field XAPLONWT.

When performing the authorization check on the entire table and authorization is granted to the requestor, the RACF/DB2 external security module returns a blank (' ') in the output field XAPLONWT and sends a return code of 0.

If the authorization check is granted on a particular column or set of columns using a generic profile, the RACF/DB2 external security module returns an asterisk ('*') in output field XAPLONWT and sends a return code of 0. DB2 provides the column name included in XAPLREL1 to the RACF/DB2 external security module.

The XAPLDIAG Output Parameter

The output parameter XAPLDIAG is used to contain return codes and reason codes. When a RACROUTE REQUEST=FASTAUTH check fails to grant access, the RACF/DB2 external security module records the failing SAF return code, RACF return and reason codes in XAPLDIAG. Each word of XAPLDIAG contains a FASTAUTH SAF return code (1 byte), the RACF return code (1 byte) and the RACF reason code (2 bytes), from left to right. All return codes and reason codes are shown in hexadecimal. In this way, DB2 or other programs have a way to trap and obtain diagnostic information.

See *z/OS SecureWay Security Server RACF Diagnosis Guide* for more information.

DB2 Aliases for System-Directed Access

RACF applies protection to the base object, not to a DB2 alias.

Considerations for Remote and Local Resources

The RACF entity check is always performed for local resources.Remote resources are always checked by the remote DB2. This also occurs when binding an application that accesses remote resources.

The WITH GRANT Option

Structured Query Language (SQL) allows authorities to be held with the WITH GRANT option, which allows users to GRANT those privileges to others. When using RACF for security authorization, this is not supported through RACF.

DB2 Object Names With Blank Characters

In DB2, it is possible to use delimited identifiers to create DB2 object names containing blank characters. However, RACF resource names cannot contain blank

characters. As a result, when the RACF/DB2 external security module encounters a DB2 object name containing blank characters, it will translate the blank characters to underscores (_, X'6D') before performing security checking. To protect DB2 objects containing blanks, you must define RACF profiles that will match an underscore (either explicitly or via generics) in place of the blank characters.

DB2 Object Names With Special Characters

In DB2, it is possible to use delimited identifiers to create DB2 object names containing any character in the EBCDIC syntactic character set. However, RACF does not allow the use of commas, semicolons, or parentheses in resource names. To protect DB2 objects containing these characters (or any other characters that are not allowed by the RACF command processors), you need to define RACF profiles containing generic characters that will match the unsupported characters.

Authority Checking for All Packages in a Collection

The naming convention for DB2 package objects is:

subsystem-name.collection-ID.package-ID.privilege-name

When a DB2 user tries to perform an operation on all packages in a collection, DB2 may pass an asterisk (*****) to the RACF/DB2 external security module in place of *package-ID*. To ensure consistent results between the RACF/DB2 external security module and the RACF command processors (SEARCH and RLIST), the asterisk (*****) in the resource name should match the asterisk (*****) in the profile name.

For example, in DB2, you can BIND a plan using all of the packages from a given collection. When that plan is subsequently executed, DB2 will check the user's authority to execute all packages in the collection by passing an asterisk (*****) in place of the collection name. For example, suppose the following DB2 commands are issued for subsystem DSN:

```
BIND PACKAGE (DSNTEP2) MEMBER(DSNTEP2) ACT(REP) ISO(CS)
BIND PLAN(DSNTEP42)  PKLIST(DSNTEP2.*) ACT(REP) ISO(CS)
RUN  PROGRAM(DSNTEP2) PLAN(DSNTEP42) -
```

When DB2 gets to the execution step, it invokes the RACF/DB2 external security module to check the user's authority to EXECUTE package DSNTEP2.*, where the asterisk (*****) means all packages in the collection.

The RACF/DB2 external security module checks the user's authority to resource:

```
DSN.DSNTEP2.*.EXECUTE     (in class MDSNPK)
```

The RACF profile name protecting this resource should contain a single asterisk (*****) to match the asterisk (*****) in the resource name.

AUTOBIND Requests for User-Defined Functions

RACF fails all authorization checks associated with AUTOBIND requests for user-defined functions. That is, when:

- XAPLAUTO (in XAPLFLG1) is non-zero,
- XAPLTYPE indicates a function ("F"), and
- XAPLPRIV is 64 (EXECUTE)

then a return code 8 and reason code 17 are returned, and no resource check is performed. This causes the AUTOBIND request to fail. A manual REBIND is then required.

Identity Used for Authorization Checks

The RACF/DB2 external security module receives user identification information in the XAPL (DSNXAPL) parameter list that is passed by DB2. In the XAPL, the RACF/DB2 external security module receives:

- a pointer to the input ACEE that represents the identity of the requester (XAPLUPRM).
- the 1–8-character user ID of the requester (XAPLUPRM).

  Note: The XAPLUPRM value is used for all RACF authorization checking, although RACF actually checks the input ACEE itself to determine this identity. The identity represented by the ACEE is the same as the user ID passed in XAPLUPRM.
- the 1–8-character authorization ID (XAPLUCHK) that DB2 uses for the authorization check. The XAPLUCHK may contain a value that is not a RACF user ID or group, and it may differ from the XAPLUPRM.

While the RACF/DB2 external security module uses the XAPLUCHK and XAPLUPRM values to perform ownership checks, it performs all access authorization checks using only XAPLUPRM.

It is possible for the XAPLUCHK value to be different from the user ID (XAPLUPRM) represented in the ACEE pointed to by XAPLACEE. For example, this can occur when a BIND request is issued and the binder is not the owner of the plan or package. The RACF/DB2 external security module is invoked to determine whether the binder is authorized to do the BIND. If this check is successful, it is then invoked to check the binder's authorization to access each DB2 resource accessed in the plan or package. For the BIND check, XAPLUPRM and XAPLUCHK have the authorization ID of the binder. However, for the subsequent checks on the DB2 resources accessed in the plan or package, XAPLUPRM still has the authorization ID of the binder, but XAPLUCHK now has the authorization ID of the plan or package owner. For the BIND to succeed, the binder must have authorization to bind this plan or package, and be authorized to access all DB2 resources accessed in it. DB2 authorization performs the subsequent checks on the owner of the plan/package and not the binder.

Administering the RACF/DB2 External Security Module

The RACF/DB2 external security module:

- Receives control from the DB2 access control authorization exit point (DSNX@XAC) to handle DB2 authorization checks
- Provides a single point of control for security administration
- Provides the ability to define security rules before a DB2 object is created
- Allows security rules to persist when a DB2 object is dropped
- Provides the ability to protect multiple DB2 objects with a single security rule using a combination of generic, grouping, and member profiles
- Eliminates DB2 *cascading revoke*
- Preserves DB2 privileges and administrative authorities
- Provides flexibility for multiple DB2 subsystems with a single set of RACF profiles
- The ability to validate a user ID before giving it access to a DB2 object

To indicate the function to be performed, DB2 passes one of three function codes to the RACF/DB2 external security module—for initialization, authorization checking, or

termination. See *z/OS SecureWay Security Server RACF System Programmer's Guide* for initialization and termination information.

Initialization

Any DB2 classes you want to use must be active during RACF/DB2 external security module initialization (XAPLFUNC = 1). You cannot activate a DB2 class later and expect the RACF/DB2 external security module to perform authorization checking against it, because the class will not be RACLISTed. RACLISTing is only done during initialization of the RACF/DB2 external security module.

To start using DB2 classes that were not previously RACLISTed during initialization, you will need to stop and re-start DB2.

Once the DB2 subsystem has initialized, the following command needs to be issued to affect profile changes for classes being used by the RACF/DB2 external security module:

```
SETROPTS RACLIST(classname) REFRESH
```

The following informational messages are issued for each initialization: IRR908I, IRR909I, IRR910I, and IRR911I.

Note: The classes listed in message IRR911I may be a valid subset of the classes listed in message IRR910I. The RACF/DB2 external security module is programmed to RACLIST all supported DB2 classes. Message IRR910I lists the DB2 classes for which the RACF/DB2 external security module has initiated RACLIST. However, message IRR911I lists only the DB2 classes that were successfully RACLISTed. In order to be successfully RACLISTed, a DB2 class must be active and contain at least one profile. Therefore, there are valid circumstances where the list of classes contained in IRR911I will be a subset of those listed in IRR910I.

Failure to Initialize

If the RACF/DB2 external security module fails to initialize for any reason, messages IRR900A, IRR901A, IRR902A, and IRR903A are issued to the security console. If this occurs, you can do the following:

1. Check that the DB2 classes are active, and that there is at least one profile defined in each class.
2. Examine RACROUTE REQUEST=LIST return and reason codes to determine why RACLISTing of classes is failing in the RACF/DB2 external security module.
3. Check if any other required resources (GETMAIN, for example) are obtainable.

Return Codes and Reason Codes From Initialization

Return codes from the RACF/DB2 external security module are returned in the DB2-supplied EXPL field EXPLRC1. Reason codes from the RACF/DB2 external security module are returned in the DB2-supplied EXPL field EXPLRC2. See *z/OS SecureWay Security Server RACF System Programmer's Guide* for the meanings of the return and reason codes from the initialization of the RACF/DB2 external security module.

Deferring to Native DB2 Authorization

Deferring to native DB2 authorization may or may not require removal of the RACF/DB2 external security module. A return code of 4 from the RACF/DB2 external security module indicates that DB2 should defer to DB2 security checking for that particular authorization check.

Removing the RACF/DB2 External Security Module

If the RACF/DB2 external security module is removed, DB2 reverts to using native DB2 authorization, in which authority is determined by DB2 catalogs.

In addition, you may need to:

1. Inactivate any classes related to the DB2 processing
2. Make the necessary GRANTs in DB2

Authorization Checking (XAPLFUNC = 2)

The RACF/DB2 external security module requires an input ACEE to perform authority checking. When an input ACEE (XAPLACEE) is not provided to the RACF/DB2 external security module, it defers to DB2 for authority checking (EXPLRC1 set to 4). See the *DB2 Administration Guide* for the requests for which the input ACEE (XAPLACEE) is set to zero. For these requests, authority checking must be implemented using the DB2 GRANT and REVOKE commands. RACF profiles defined for these requests will *not* be used.

The RACF/DB2 external security module performs FASTAUTH checks during authorization according to the rule sequences described in “Appendix D. RACF/DB2 External Security Module: Authorization Checking” on page 595. In DB2, there is no concept of negative access level. RACF/DB2 external security module processing ends when FASTAUTH returns a return code of 0 or the list of checks for the request has been exhausted. Failure audit records are only created for the first failing resource. All audit records associated with the same invocation of the exit contain the same LOGSTR data. See “Authorization Processing: Examples” on page 408 for examples.

Return and Reason Codes

The following return and reason codes are shown in decimal notation.

Return Code	Meaning	
0	Access permitted	
	Reason Code	**Meaning**
	0	Access permitted by FASTAUTH checking.
	13	Access permitted by implicit privilege of ownership.
	14	Access permitted because current SQL ID matches schema name.
4	Unable to determine; perform DB2 authorization checking	
	Reason Code	**Meaning**
	0	Input class (XAPLTYPE) not active.
	11	Input ACEE (XAPLACEE) not provided.
	14	The ALET could not be created for cross memory ACEE.
	15	Input privilege code (XAPLPRIV) or input class (XAPLTYPE) not defined to the RACF/DB2 external security module.
	16	Input privilege code (XAPLPRIV) does not contain any rules.
8	Access denied	

Reason Code	Meaning
17	Autobind indicator (XAPLAUTO) is not zero indicating AUTOBIND was requested. Manual REBIND is required.

FASTAUTH Return Code Translation

Each time the RACF/DB2 external security module is invoked, it may in turn invoke RACROUTE REQUEST=FASTAUTH multiple times. If one of the FASTAUTH requests completes with a return code of zero, the return code passed back to DB2 will be zero. If none of the FASTAUTH requests complete with a return code of zero, the collection of return codes from FASTAUTH must be translated into a single resultant return code. Return code translation can be summarized as follows:

If all object resource checks result in a return code of 4 and none of the DSNADM checks result in a return code of 0, the RACF/DB2 external security module passes back a return code of 4.

If at least one object resource check results in a return code of 8 and none of the DSNADM checks result in a return code of 0, the RACF/DB2 external security module passes back a return code of 8.

If no object resource profiles are checked and all of the DSNADM checks result in a return code of 8, the RACF/DB2 external security module will pass back a return code of 8. Otherwise, if no object resources are checked and the DSNADM checks result in a mix of 4s and 8s, the RACF/DB2 external security module passes back a return code of 4.

All failing SAF/RACF return codes and RACF reason codes are placed in the output parameter field in XAPLDIAG, to be returned to DB2. This information is then available to DB2, SQL, or other programs to obtain diagnostic information from it.

Table 32 illustrates the method used to do this translation.

Table 32. FASTAUTH Return Code Translation

Return Code from Object Profile	Return Code from ADM Profile	Output Return Code
None	All 4s	04
None	All 8s	08
None	Mix	04
All 4s	None	04
All 4s	All 4s	04
All 4s	All 8s	04
All 4s	Mix	04
All 8s	None	08
All 8s	All 4s	08
All 8s	All 8s	08
All 8s	Mix	08
Mix	None	08
Mix	All 4s	08
Mix	All 8s	08
Mix	Mix	08

Authorization Processing: Examples

- Examples 1 through 4 show authority checks performed on tables using supplied classes for multiple-subsystem scope (classification model 2).
- Example 5 shows authority checks performed on tables using installation-defined classes for multiple-subsystem scope (classification model 2).
- Example 6 shows authority checks performed on tables using installation-defined classes for single-subsystem scope (classification model 1).

Example 1: Allowing Access (Auditing for Failures)

This example shows how the RACF/DB2 external security module allows access to a DB2 object (a table) based on a DB2 administrative authority profile. Auditing is activated for failures.

In this example, user ID MIKEJ is trying to alter a table called BDA0828.EMP in database JBW2000.

Setup

- Classification model (&CLASSOPT): `2`
- Class name root (&CLASSNMT): `DSN`
- Class name suffix (&CHAROPT): `1`

  This is the default value, but it is not used with supplied classes.
- DB2 subsystem name: `VHH1`
- Profiles:
  - Defined in the MDSNTB class:

    ```
    VHH1.BDA0828.EMP.ALTER
    - AUDIT(FAILURES(READ))
    - UACC(NONE)
    ```
  - Defined in the DSNADM class:

    ```
    VHH1.SYSADM
    - AUDIT(FAILURES(READ))
    - UACC(NONE)
    - ID(MIKEJ) ACCESS(READ)
    ```
- User ID MIKEJ has SYSADM authority in DB2

Profile Checking

RACF checks the following resources:

1. VHH1.BDA0828.EMP.ALTER in class MDSNTB

   Results:
   - Access is denied (return code 8)
   - No failure message (ICH408I) is issued
   - No audit records are created
2. VHH1.JBW2000.DBADM in class DSNADM

   Results:
   - No profile is found (return code 4)
   - No failure message (ICH408I) is issued
   - No audit records are created
3. VHH1.SYSCTRL in class DSNADM

   Results:

- No profile is found (return code 4)
- No failure message (ICH408I) is issued
- No audit records are created

4. VHH1.SYSADM in class DSNADM

   Results:
   - Access is granted (return code 0)
   - No failure message (ICH408I) is issued
   - No audit records are created

Final Results

1. The RACF/DB2 external security module sends a return code of 0 to DB2.
2. DB2 issues a successful DSNT400I message to the output log.

Example 2: Allowing Access (Auditing for All Attempts)

This example shows how the RACF/DB2 external security module allows access to a DB2 object (a table) based on a DB2 administrative authority profile. Auditing is activated for all access attempts.

In this example, user ID MIKEJ is trying to alter a table called BDA0828.EMP in database JBW2000.

Setup

- Classification model (&CLASSOPT): `2`
- Class name root (&CLASSNMT): `DSN`
- Class name suffix (&CHAROPT): `1`

  This is the default value, but it is not used with supplied classes.
- DB2 subsystem name: `VHH1`
- Profiles:
  - Defined in the MDSNTB class:

    ```
    VHH1.BDA0828.EMP.ALTER
    - AUDIT(ALL(READ))
    - UACC(NONE)
    - ID(MIKEJ) ACCESS(NONE)
    ```
  - Defined in the DSNADM class:

    ```
    VHH1.SYSADM
    - AUDIT(ALL(READ))
    - UACC(NONE)
    - ID(MIKEJ) ACCESS(READ)
    ```
- User ID MIKEJ has SYSADM authority in DB2

Profile Checking

RACF checks the following resources:

1. VHH1.BDA0828.EMP.ALTER in class MDSNTB

   Results:
   - Access is denied (return code 8)
   - No failure message (ICH408I) is issued
   - No audit records are created
2. VHH1.JBW2000.DBADM in class DSNADM

   Results:
   - No profile is found (return code 4)
   - No failure message (ICH408I) is issued

- No audit records are created

3. VHH1.SYSCTRL in class DSNADM

   Results:
   - No profile is found (return code 4)
   - No failure message (ICH408I) is issued
   - No audit records are created

4. VHH1.SYSADM in class DSNADM

   Results:
   - Access is granted (return code 0)
   - No failure message (ICH408I) is issued
   - An audit record is created, which includes the following log string data:
     - The BDA0828.EMP.ALTER profile name
     - Input parameters identifying the request from DB2

Final Results

1. The RACF/DB2 external security module sends a return code of 0 to DB2.
2. DB2 issues a successful DSNT400I message to the output log.

Example 3: Denying Access

This example shows how the RACF/DB2 external security module denies to a DB2 object (a table). Auditing is activated for all access attempts.

In this example, user ID MIKEJ is trying to alter a table called BDA0828.EMP in database JBW2000.

Setup

- Classification model (&CLASSOPT): `2`
- Class name root (&CLASSNMT): `DSN`
- Class name suffix (&CHAROPT): `1`

  This is the default value, but it is not used with supplied classes.
- DB2 subsystem name: `VHH1`
- Profiles:

  Defined in the MDSNTB class:

  ```
  VHH1.SYSADM
  VHH1.BDA0828.EMP.ALTER
  – AUDIT(ALL(READ))
  – UACC(NONE)
  – ID(MIKEJ) ACCESS(NONE)
  ```
- User ID MIKEJ has SYSADM authority in DB2

Profile Checking

RACF checks the following resources:

1. VHH1.BDA0828.EMP.ALTER in class MDSNTB

   Results:
   - Access is denied (return code 8)
   - No failure message (ICH408I) is issued
   - No audit records are created

2. VHH1.JBW2000.DBADM in class DSNADM

   Results:
   - No profile is found (return code 4)
   - No failure message (ICH408I) is issued

- No audit records are created

3. VHH1.SYSCTRL in class DSNADM

   Results:
   - No profile is found (return code 4)
   - No failure message (ICH408I) is issued
   - No audit records are created

4. VHH1.SYSADM in class DSNADM

   Results:
   - No profile is found (return code 4)
   - No failure message (ICH408I) is issued
   - No audit records are created

5. VHH1.BDA0828.EMP.ALTER in class MDSNTB

   Results:
   - Access is denied (return code 8)
   - Failure message (ICH408I) is issued
   - An audit record is created, which includes the following log string data:
     - The BDA0828.EMP.ALTER profile name
     - Input parameters identifying the request from DB2

Final Results

1. The RACF/DB2 external security module sends a return code of 8 to DB2.
2. DB2 issues a DSNT408I message to the output log.

Example 4: Deferring to DB2

This example shows how the RACF/DB2 external security module defers to DB2, because the DB2 object (a table) is not protected by RACF.

In this example, user ID MIKEJ is trying to alter a table called BDA0828.EMP in database JBW2000.

Setup

- Classification model (&CLASSOPT): `2`
- Class name root (&CLASSNMT): `DSN`
- Class name suffix (&CHAROPT): `1`

  This is the default value, but it is not used with supplied classes.
- DB2 subsystem name: `VHH1`
- Profiles:
  - `VHH1.BDA0828.EMP.ALTER`
  - `VHH1.SYSADM`
    `- AUDIT(ALL(READ))`
- User ID MIKEJ has SYSADM authority in DB2

Profile Checking

RACF checks the following resources:

1. VHH1.BDA0828.EMP.ALTER in class MDSNTB

   Results:
   - No profile is found (return code 4)
   - No failure message (ICH408I) is issued
   - No audit records are created

2. VHH1.JBW2000.DBADM in class DSNADM

Results:
- No profile is found (return code 4)
- No failure message (ICH408I) is issued
- No audit records are created

3. VHH1.SYSCTRL in class DSNADM

   Results:
   - No profile is found (return code 4)
   - No failure message (ICH408I) is issued
   - No audit records are created

4. VHH1.SYSADM in class DSNADM

   Results:
   - No profile is found (return code 4)
   - No failure message (ICH408I) is issued
   - No audit records are created

Final Results

1. The RACF/DB2 external security module sends a return code of 4 to DB2.
2. DB2 issues a successful DSNT400I message to the output log, MIKEJ has SYSADM authority in DB2.

Example 5: Allowing Access (Multiple-Subsystem Scope)

This example shows how the RACF/DB2 external security module allows access to a DB2 object (a table) based on a DB2 administrative authority profile. The installation has defined classes MSLH1TB1 and SLH1ADM1. Auditing is activated for all access attempts.

In this example, user ID MIKEJ is trying to alter a table called BDA0828.EMP in database JBW2000.

Setup

- Classification model (&CLASSOPT): `2`
- Class name root (&CLASSNMT): `SLH1`
- Class name suffix (&CHAROPT): `1`
- DB2 subsystem name: `VHH1`
- Profiles:

  Defined in the MSLH1TB1 class:

  ```
  VHH1.BDA0828.EMP.ALTER
  – AUDIT(ALL(READ))
  – UACC(NONE)
  ```

  Defined in the SLH1ADM1 class:

  ```
  VHH1.SYSADM
  – AUDIT(ALL(READ))
  – UACC(NONE)
  – ID(MIKEJ) ACCESS(READ)
  ```

- User ID MIKEJ has SYSADM authority in DB2

Profile Checking

RACF checks the following resources:

1. VHH1.BDA0828.EMP.ALTER in class MSLH1TB1

   Results:

- Access is denied (return code 8)
- No failure message (ICH408I) is issued
- No audit records are created

2. VHH1.JBW2000.DBADM in class SLH1ADM1

   Results:
   - No profile is found (return code 4)
   - No failure message (ICH408I) is issued
   - No audit records are created

3. VHH1.SYSCTRL in class SLH1ADM1

   Results:
   - No profile is found (return code 4)
   - No failure message (ICH408I) is issued
   - No audit records are created

4. VHH1.SYSADM in class SLH1ADM1

   Results:
   - Access is granted (return code 0)
   - No failure message (ICH408I) is issued
   - An audit record is created, which includes the following log string data:
     - The BDA0828.EMP.ALTER profile name
     - Input parameters identifying the request from DB2

Final Results

1. The RACF/DB2 external security module sends a return code of 0 to DB2.
2. DB2 issues a successful DSNT400I message to the output log.

Example 6: Allowing Access (Single-Subsystem Scope)

This example shows how the RACF/DB2 external security module allows access to a DB2 object (a table) based on a DB2 administrative authority profile. The installation has defined classes MVHH1TB1 and VHH1ADM1. Auditing is activated for all access attempts.

In this example, user ID MIKEJ is trying to alter a table called BDA0828.EMP in database JBW2000.

Setup

- Classification model (&CLASSOPT): `1`
- Class name root (&CLASSNMT): `DSN`

  This is the default value, but it is not used in single-subsystem scope.
- Class name suffix (&CHAROPT): `1`
- DB2 subsystem name: `VHH1`
- Profiles:

  Defined in the MVHH1TB1 class:

```
BDA0828.EMP.ALTER
– AUDIT(ALL(READ))
– UACC(NONE)
```

  Defined in the VHH1ADM1 class:

```
SYSADM
– AUDIT(ALL(READ))
– UACC(NONE)
```

  - ID(MIKEJ) ACCESS(READ)
- User ID MIKEJ has SYSADM authority in DB2

Profile Checking

RACF checks the following resources:

1. BDA0828.EMP.ALTER in class MVHH1TB1

   Results:
   - Access is denied (return code 8)
   - No failure message (ICH408I) is issued
   - No audit records are created
2. JBW2000.DBADM in class VHH1ADM1

   Results:
   - No profile is found (return code 4)
   - No failure message (ICH408I) is issued
   - No audit records are created
3. SYSCTRL in class VHH1ADM1

   Results:
   - No profile is found (return code 4)
   - No failure message (ICH408I) is issued
   - No audit records are created
4. SYSADM in class VHH1ADM1

   Results:
   - Access is granted (return code 0)
   - No failure message (ICH408I) is issued
   - An audit record is created, which includes the following log string data:
     - The BDA0828.EMP.ALTER profile name
     - Input parameters identifying the request from DB2

Final Results

1. The RACF/DB2 external security module sends a return code of 0 to DB2.
2. DB2 issues a successful DSNT400I message to the output log.

Converting DB2 Authorizations to RACF Profiles

You can convert the contents of the SYSIBM.SYS*xxx*AUTH tables to equivalent RACF profiles. These RACF profiles are used with the RACF/DB2 external security module. A sample conversion tool called RACFDB2 may assist you to build these RACF profiles. This tool is described in *Ready for e-business: OS/390 Security Server Enhancements*, an IBM systems center publication. For information on how to get this tool and others from the RACF home page or via anonymous FTP, see “Internet Sources” on page xxv.

Common Problems and Considerations

Common problems that could occur as a result of defining special classes in the installation-defined class descriptor table (ICHRRCDE) follow:

- A class is not defined in the class descriptor table (CDT).

  This results in a return code of 4 (profile not found) from the RACF/DB2 external security module.
- Incorrect linkage editor procedures from either the CDT or the router table.
- A class is defined in the CDT but is not defined in the router table.
- Classes are defined and link-edited properly, but a re-IPL had not occurred to pick up the changes.

- Single-subsystem scope class names are being used and a new subsystem is using the RACF/DB2 external security module before classes for the subsystem have been defined.
- Messages IRR900A, IRR901A, IRR902A, and IRR903A are issued because the RACF/DB2 external security module could not initialize correctly.
  1. Check to see if DB2 classes are active.
  2. Determine if and why RACLISTing of classes is failing in exit by examining RACROUTE REQUEST=LIST return and reason codes.
  3. Check to see if any other required resources (such as GETMAIN, for example) are obtainable.

DB2

Chapter 12. RACF and DCE

This chapter describes using RACF with z/OS DCE or OS/390 DCE.

Attention

Read *z/OS DCE Introduction* if you need an overview of DCE technology and terminology.

Also, see “z/OS UNIX Application Considerations” on page 514 in Chapter 18. RACF and z/OS UNIX.

The interoperation of RACF with DCE enables DCE application servers on z/OS or OS/390 to map a DCE user identity *(principal)* to a RACF user ID. The mapping of a DCE principal to a RACF user ID is known as *cross linking.* The identity cross linking information contained in RACF can be used by:

- z/OS DCE and OS/390 DCE to determine which users are eligible for single signon to DCE
- Application servers residing on z/OS or OS/390 to determine the RACF user ID of clients

Cross Linking DCE Identities and RACF User IDs

By supporting z/OS DCE and OS/390 DCE, RACF establishes a *cross linking* of identity between a RACF user ID and a DCE principal. This support includes the DCE segment and the DCEUUIDS class, which define profiles that link a DCE universal unique identifier (UUID) to a RACF user ID. Figure 45 on page 418 shows these relationships.

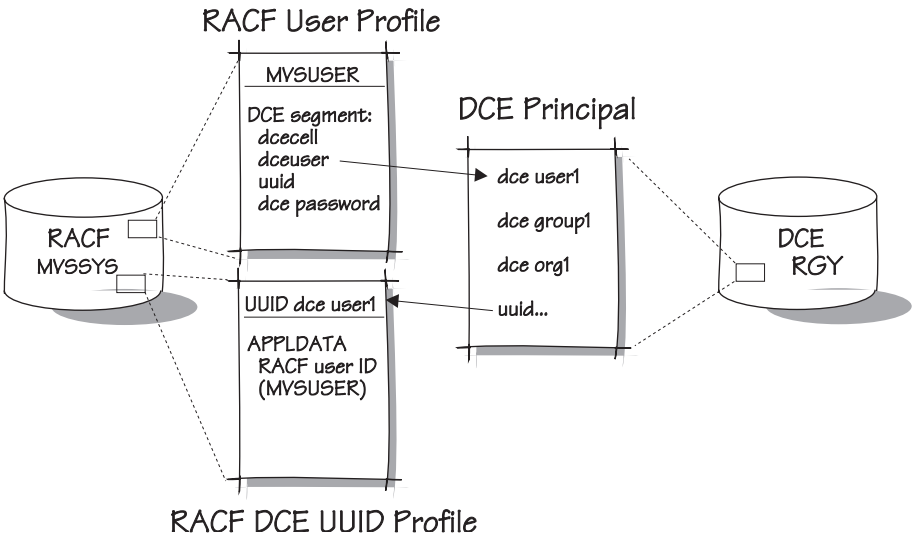


Figure 45. DCE/RACF User ID Cross Linking Example

Between the DCE segment associated with a RACF user profile and the DCEUUIDS class profile, a RACF identity is *cross linked* to its corresponding DCE identity.

Defining Cross Linking Information

Two DCE utilities, `mvsimpt` and `mvsexpt`, administer DCE information in the RACF database and create the initial cross linking information between the RACF user profile and the DCE principal registry using the import and export processes.

Import process
: is the process of defining and updating users to the DCE registry that were previously defined to RACF

Export process
: is the process of defining and updating users to RACF that were previously defined to the DCE registry

The utilities can be started with information from either the RACF database or DCE registry:

mvsimpt is a two-pass utility that creates DCE principal entries in the DCE registry for the set of RACF users chosen to be cross linked, based on the output from the RACF database unload utility. The unloaded RACF database is sorted by the administrator according to RACF user IDs with a RACF DCE segment and filtered by the utility according to processed entries from previous `mvsimpt` and `mvsexpt` processing.

mvsexpt is a two-pass utility that cross populates a set of DCE principals with a RACF database. It creates and updates the RACF DCE segment for each DCE principal being cross linked with the RACF database. `mvsexpt` takes a specified input file or the DCE registry for each principal specified and creates the RACF DCE segment and the RACF general resource class, DCEUUIDS.

For more information on these utilities, see *z/OS DCE Administration Guide*.

The RACF DCEUUIDS Class

RACF uses the general resource class, DCEUUIDS, which contains the cross linking information for each RACF/DCE user.

As shown on Figure 45 on page 418, profiles defined to the RACF DCEUUIDS class associate a DCE principal with a RACF user ID on a particular system that is part of a DCE cell. The profile name is the string form of the DCE cell UUID and principal UUID. The APPLDATA field of this profile contains the RACF user ID. Profiles defined to the DCEUUIDS class should include the cell's UUID. The general form of the profiles defined to this class is:

cell-string-form-uuid.principal-string-form-uuid

Using the DCE segment and the DCEUUIDS class profile, applications using a new SAF callable service, extensions to the kernel, or extensions to the C function library can determine a user's DCE or RACF identity. Ensure that DCE segment definitions and DCEUUIDS class profiles include the cell UUID.

Defining Profiles to the RACF DCEUUIDS Class

Whenever DCE information is updated in a USER profile with the RACF ISPF panels or one of the following commands:

- ALTUSER
- DELUSER
- ADDUSER

the corresponding DCEUUIDS profiles will automatically be updated. When RACF creates a DCEUUIDS profile as a result of an ADDUSER or ALTUSER command, the user ID of the command issuer becomes the owner of the DCEUUIDS profile.

In addition, profiles defined to the DCEUUIDS class can be created and maintained by using DCE import and export utilities.

Activating the DCEUUIDS Class

Before profiles defined to the DCEUUIDS class can be used, the class must be activated. To activate the DCEUUIDS class enter:

```
SETROPTS CLASSACT(DCEUUIDS)
```

Administering DCE Information in RACF

Updating a user's principal and home cell DCE UUIDs results in automatic updates to the identity mapping profiles contained in the DCEUUIDS class.

Example:

As RACF administrator, you are changing a RACF-defined user who was previously processed by the DCE utilities to correct an initial data entry error. Also, the DCEUUIDS class profile for this user was deleted inadvertently.

The existing RACF user ID, `CSMITH`, has been cross-linked with the DCE principal `charles`.

This DCE user has been assigned a DCE principal UUID of `004386ea-ebb6-1ec3-bcae-10005ac90feb` and a DCE principal name of `charles`. This DCE user will

become a principal of the mvssys DCE cell. The UUID for the mvssys DCE cell is 003456ab-ecb7-7de3-ebda-95531ed63dae.

```
ALTUSER CSMITH +
DCE(UUID(004386ea-ebb6-1ec3-bcae-10005ac90feb) +
     DCENAME(charles) HOMECELL(/.../mvssys.endicott.ibm.com) +
     HOMEUUID(003456ab-ecb7-7de3-ebda-95531ed63dae))
```

Figure 46. Changing a DCE user with ALTUSER

You should list the DCE segment information for the RACF user ID CSMITH to ensure that no errors were made entering the data for the ALTUSER command shown in the example. The corresponding DCE mapping profile in the DCEUUIDS class is updated to reflect the new information automatically.

```
LISTUSER CSMITH NORACF DCE
USER=CSMITH
DCE INFORMATION
---------------
 UUID= 004386ea-ebb6-1ec3-bcae-10005ac90feb
 DCENAME= charles
 HOME CELL UUID= 003456ab-ecb7-7de3-ebda-95531ed63dae
 HOME CELL= /.../mvssys.endicott.ibm.com
 DCE AUTOLOGIN= NO
```

Figure 47. Output of the LISTUSER Command for user CSMITH

Single Signon Support for DCE

RACF support provides for a *single signon to DCE* feature. DCE single signon logs z/OS and OS/390 users into DCE automatically if those users have already been authenticated by RACF. The single signon support is *not* intended to be used by application servers. Single signon support should be enabled only for end users.

To start single signon to DCE processing, the following conditions must be met:
- You have requested single signon to DCE processing.
- The user is not currently logged into DCE.
- The user invokes a DCE application.
- The user is defined as a DCE principal to the DCE registry.

Before DCE single signon support can be invoked for a z/OS or OS/390 user, the user must be enrolled for the *single signon to DCE* feature. To enroll:

1. RACF setup procedures for DCE interoperability must be completed.
2. A populated DCE segment must be created for the user in the RACF user profile. The user profile DCE segment must contain the user's DCE information. IBM recommends that you use the DCE utility mvsimpt and the DCE utilities mvsimpt and mvsexpt as an administrative aid.
3. The AUTOLOGIN value in the user's DCE segment must be set to YES to invoke single signon processing.
4. The user must have saved the current DCE password in the RACF DCE segment by invoking the DCE storepw command.

Attention

- You still need to maintain a separate password for RACF and DCE.
- You must use the DCE `storepw` to keep current your DCE password that is stored in RACF. If you change your DCE password that is known to the DCE registry, you must use the `storepw` command to save your new DCE password in RACF.
- See *z/OS DCE Administration Guide* for more information on single signon restrictions.
- See *z/OS DCE Command Reference* for a discussion of the DCE `storepw` command.

The RACF KEYSMSTR Class

The RACF KEYSMSTR class is a general resource class that contains the DCE.PASSWORD.KEY profile. This profile holds the encryption key that is used for encrypting and decrypting a user's DCE password for use in DCE single signon support.

Attention

- Before a user can save a DCE password in the RACF database or before the DCE single signon feature can be used:
  - The KEYSMSTR class profile DCE.PASSWORD.KEY must be created with the SSIGNON segment.
  - The RACF KEYSMSTR class must be activated.
- If a cryptographic product is not on the system, you must specify the KEYMASKED sub-operand on the SSIGNON operand of the RDEFINE or RALTER command when defining or altering the KEYSMSTR profile.
- If a cryptographic product is present on the system, you can specify the KEYENCRYPTED sub-operand on the SSIGNON operand of the RDEFINE or RALTER command. If you specify the KEYENCRYPTED sub-operand, the cryptographic product must be active when you define the profile to the KEYSMSTR class.

Note: When using the secured signon facilities with encryption, some of the Integrated Cryptographic Service Facility (ICSF) modules must be put in the link pack area (LPA) or the modified link pack area (MLPA) so these modules can be accessed from RACF. The modules that must be put in the LPA or MLPA are:

CSNBCKI
CSNBDEC
CSNBENC
CSNBKRC
CSNBKRD
CSNBKRW

Depending on the release of ICSF, some of these modules may not exist. RACF checks ICSF and only uses existing modules.

Defining the RACF KEYSMSTR Class Profile

The profile defined to the KEYSMSTR class contains a SSIGNON segment that holds either the masked or encrypted value for the key that is used to encrypt DCE passwords stored in the RACF database.

The profile defined to the KEYSMSTR class that contains the encryption key is:

```
DCE.PASSWORD.KEY
```

Example:

In this example:

- Users' DCE passwords are masked using a key saved in the KEYSMSTR class DCE.PASSWORD.KEY.
- The key is 0034639986ACCFDE.

> **Attention**
> For security reasons, you should choose a key that is known only to you, the system security administrator.

To define the profile, enter:

```
RDEFINE KEYSMSTR DCE.PASSWORD.KEY +
  SSIGNON(KEYMASKED(0034639986ACCFDE))
```

If you list the DCE.PASSWORD.KEY profile's SSIGNON segment, RLIST shows output that the SSIGNON information is keymasked and not displayable.

Activating the KEYSMSTR Class

Before the DCE.PASSWORD.KEY profile can be used by RACF, the KEYSMSTR class must be activated. To activate the KEYSMSTR class enter:

```
SETROPTS CLASSACT(KEYSMSTR)
```

Chapter 13. RACF and Tivoli Products

RACF provides interfaces to allow profiles on the RACF database to be managed through the Tivoli User Administration and Tivoli Security Management products. See the Tivoli product documentation for details on managing RACF profiles using these products.

Establishing a RACF Identity for a Tivoli Administrator

A RACF identity must be established for a Tivoli administrator so this administrator can manage security on z/OS and OS/390 endpoints. The requests to update RACF profile information will run under the authority of the RACF user ID associated with the Tivoli administrator.

Define a general resource profile in the TMEADMIN class to associate a Tivoli administrator with a RACF user ID. The profile name is the administrator's Tivoli principal name. If Kerberos is in use, the Tivoli principal name is the Kerberos principal name. If Kerberos is not in use, the Tivoli principal name is a combination of the UNIX login name and host name used to access the Tivoli command language interface (CLI) or graphical user interface (GUI). The format for the Tivoli principal name (when Kerberos is not in use) is:

unix-login-name@host-name

Note: The Tivoli principal name is displayed in the title bar of the Tivoli desktop. You may also issue the following command to determine which principal name Tivoli uses when you issue a Tivoli command or start the GUI from the current shell:

```
objcall 'objcall 0.0.0 get_oserv' o_get_principal
```

For more information about the Tivoli principal name, see *Tivoli Framework Planning and Installation Guide*.

The APPLDATA field of this profile must contain the RACF user ID. It is recommended that each Tivoli administrator user ID map to one and only one RACF user ID. The sharing of a RACF user ID by multiple Tivoli administrators is not recommended. Therefore, a profile in the TMEADMIN class should exist for each Tivoli administrator who is able to perform RACF administration.

For example, the Tivoli administrator `jdiamond` in the Tivoli Management Region `pok01` has a RACF user ID of `JOHND`. To define a general resource profile in the TMEADMIN class to associate `jdiamond` in the Tivoli Management Region `pok01` with the RACF user ID `JOHND`, a RACF administrator with SPECIAL authority would issue:

```
RDEFINE TMEADMIN jdiamond@pok01 APPLDATA('JOHND')
```

The RACF user IDs used by Tivoli administrators must have the appropriate RACF authority to manage the set of resources these administrators are responsible for. Unless the set of managed RACF resources is very limited, the RACF user ID will require the SPECIAL attribute. To manage the global audit setting in resource profiles, the RACF user ID will require the AUDITOR attribute.

Listing Profiles in the TMEADMIN Class

You can list a TMEADMIN class profile using the RLIST command.

For example, to list a TMEADMIN class profile for Tivoli administrator `jdiamond` in the Tivoli Management Region `pok01`, enter:

```
RLIST TMEADMIN jdiamond@pok01
```

You will see the following output:

```
RLIST TMEADMIN jdiamond@pok01

CLASS      NAME
-----      ----
TMEADMIN   JDIAMOND@POK01

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
 00    IBMUSER         NONE              ALTER    NO

INSTALLATION DATA
-----------------
NONE

APPLICATION DATA
----------------
JOHND

AUDITING
--------
FAILURES(READ)

NOTIFY
------
NO USER TO BE NOTIFIED
```

Figure 48. RLIST Output of a TMEADMIN Class Profile

Chapter 14. RACF and Information Management System (IMS)

This chapter describes factors that Information Management System (IMS) owners and administrators should consider when using IMS and RACF. Note that many of the names used in the scenarios are arbitrary. The names you use for user IDs, group names, and other such items will differ, and the procedures that you decide to follow may vary from those given as examples here.

Overview of RACF and IMS

IMS uses RACF facilities to:

- Provide individual user (terminal operator) identification and authentication
- Control access through operator identification:
  - Control access to IMS physical terminals
  - Control access to IMS transactions
  - Control access to IMS control regions
  - Control access to control region resources (PSBs, transactions, and logical terminal names) for message processing regions and batch message processing regions
- Delegate the ability to define users, protected transactions, and transaction lists to people outside the data processing department, if desired
- Provide an audit trail of individual operator actions, including database updates, on the IMS log
- Provide an operator identifier to application programs (through the I/O PCB)
- Control installation-specific resource access by using the AUTH call in IMS application programs

You should be familiar with the IMS system generation process and the other security features of IMS, such as password, logical terminal, and transaction security.

The scenarios in this chapter describe how to use RACF to control access to IMS resources. The order of the scenarios is, generally, from the easiest and most cost-effective to the more complex.

Controlling Access to IMS System Data Sets and Databases

As described earlier in this publication, RACF can control access to resources such as data sets, tape volumes, transactions, and so forth. Many of the IMS system resources fall into the DATASET class. Examples of these resources are the data sets that comprise the IMS databases and libraries.

If access to the IMS database and library resources is not controlled, further access control within IMS is of little value because the controls within IMS depend on these resources.

It is reasonable, then, to use RACF to control access to these data sets before you use RACF to control access to resources such as transactions or terminals. In doing this, it is important to understand that many of the accesses to IMS data sets and libraries are made by IMS itself. That is, to RACF, the IMS control region is a user.

If IMS runs as a started procedure, you must assign a RACF user ID to the procedure and do one of the following:

- Add the name as an entry in the STARTED class. (This is the preferred method.)
- Add the procedure's name to the RACF started procedures table (RACF module name ICHRIN03), unless the started procedures table has already been modified at your installation to contain a generic entry.

You should assign user IDs with the PROTECTED attribute to the IMS started procedures. For more information, see "Using Started Procedures" on page 143.

Before the security or group administrator who has IMS responsibility defines the IMS control region in the STARTED class or in the started procedures table, this person *must* define IMS as a user to RACF.

```
ADDGROUP IMSPROD OWNER(IMSADMIN)
ADDUSER  IMS GROUP(IMSPROD) OWNER(IMSADMIN)
```

These two RACF commands build a RACF profile for a group named IMSPROD and a profile for a user whose identifier is IMS and who is a member of the group IMSPROD. In both cases, the OWNER of these profiles is a user called IMSADMIN. (The owner of a RACF profile is allowed to issue other RACF commands that affect the profile. The owner can alter or delete the profile.)

To protect the IMS libraries, a person with appropriate RACF authority can issue the following type of RACF commands:

```
ADDSD ('IMSPROD.RESLIB','IMSPROD.PROCLIB','IMSPROD.ACBLIB', ...)
      UACC(NONE) AUDIT(ALL) OWNER(IMSADMIN)

PERMIT 'IMSPROD.RESLIB' ID(IMS,appropriate-users-or-groups) ACCESS(READ)
PERMIT 'IMSPROD.RESLIB' ID(appropriate-users-or-groups) ACCESS(UPDATE)
```

A UACC of READ says that anyone can read the data set. A UACC of NONE says that, to do anything with the data set, the individual (or a group to which the individual belongs) must be in the access list for the data set with the appropriate level of permission. In order to prohibit unauthorized parties from opening the IMS system libraries, you should specify NONE as the UACC for these data sets.

IMS libraries (such as RESLIB, PROCLIB, ACBLIB, FORMAT, MATRIX, and JOBS), and IMS system data sets (such as QBLKS, SHMSG, and LGMSG) should be

RACF-protected with access granted to IMS (or to the group to which IMS belongs) at the appropriate level, and to the people whose job it is to maintain these libraries and system data sets.

After access to the IMS system data sets has been controlled to an acceptable level, the next logical step is to control access to the IMS database data sets. This can be done in the same manner as described for the IMS system data sets or access to these data sets can be controlled with generic profiles.

A generic profile provides an access list for a large number of resources (for example, all data sets whose first two qualifiers are IMS.PROD).

As the appropriate security or group administrator, you can issue RACF commands similar to the following to establish a generic profile:

```
ADDSD 'IMS.PROD.*' OWNER(IMSADMIN) AUDIT(ALL) UACC(NONE|READ)

PERMIT 'IMS.PROD.*' ID(IMS,appropriate-users-or-groups) ACCESS(CONTROL)
PERMIT 'IMS.PROD.*' ID(appropriate-users-or-groups) ACCESS(READ)
```

Issuing the RACF PERMIT command for the user IMS with an access level of CONTROL is necessary because IMS issues a VSAM VERIFY for its VSAM data sets. (VERIFY requires a CONTROL level of access.) For OS data sets, IMS opens the data set at the highest intent in the processing option for the PSB used. A CONTROL level of access for an OS data set is equivalent to UPDATE. Batch IMS checks the processing intent of the PSB to be used, and, if it is READ, requests an open at the READ level.

Note: For data set protection to be complete, the catalogs for the IMS system libraries and data sets and the database data sets must be RACF-protected at the appropriate level.

The steps outlined above provide a great deal of control for a small amount of effort. No IMS system generation or utility runs are needed to achieve this important degree of control.

IMS System Generation Considerations

To have IMS use RACF to control access to the IMS resources, it is necessary to go through an IMS system generation process. Regardless of the type of security chosen, values must be supplied in the IMSCTRL macro and the SECURITY operand of that macro.

In the IMSCTRL macro, you should specify an IMSID value to indicate the name of a specific IMS control region if you plan to use any form of resource access control. The default value for IMSID is `IMSA.` If multiple IMS control regions share a common RACF database, the IMSID value must be unique for each control region.

In the SECURITY operand of the IMSCTRL macro, you should specify an RCLASS value. The default value supplied by IMS is `IMS.` IMS uses this value to identify which RACF resource classes belong to a specific IMS control region. For example, if the default (`IMS`) is taken, the RACF resource classes for that control region are:

TIMS for the transaction class
GIMS for the group transaction class
AIMS for the application resource class.

If RCLASS=ABC had been specified, the RACF resource classes for that control region would be:

TABC for transactions
GABC for resource grouping profiles for transactions
AABC for the application resource class.

IMS uses these class names to perform checks for resource access. For example, with transaction authorization active, IMS issues RACROUTE REQUEST=FASTAUTH when a transaction is received. On RACROUTE REQUEST=FASTAUTH, IMS requires that the user who issued the `/SIGN ON` command have READ access authority to the specified resource.

By making the class names unique for each IMS control region, it is possible to have the same transaction name on a production system and a test system, yet have different access lists for each of them with no ambiguity. Table 33 shows this relationship.

The class names must be defined in IMS as outlined above, and they must also be defined to RACF. RACF supplies the following IMS-related classes:

Table 33. IMS Class Names, How They Are Specified, and Their Usage

Default Class Names	When RCLASS=xxx Is Specified	Usage
AIMS	Axxx	Application (AGN)
CIMS	Cxxx	Command
DIMS	Dxxx	Grouping class for command
FIMS	Fxxx	Field (in database segment)
GIMS	Gxxx	Grouping class for transaction
HIMS	Hxxx	Grouping class for field
OIMS	Oxxx	Other
PIMS	Pxxx	Database
QIMS	Qxxx	Grouping class for database
SIMS	Sxxx	Segment in database
TIMS	Txxx	Transaction (trancode)
UIMS	Uxxx	Grouping class for segment
WIMS	Wxxx	Grouping class for other

If your installation wishes to define any other class names, add the classes to the installation-defined class descriptor table (ICHRRCDE). For more information, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Notes:

1. By the IMS system generation process, the RACF resource class name is determined by concatenating the first letter of the resource classes shown above with the RCLASS value specified in the SECURITY macro in the IMS generator. If no value is specified, the default `IMS` is used. These default class names are shipped with RACF.
2. The IMS resource classes that you define should be created with the same attributes as the supplied IMS classes.

Activating the IMS classes in RACF has no effect on IMS until options are specified in the IMS system and IMS is started with the options in effect.

Establishing Audit Trail Capabilities

Before you control access to any IMS internal resources, it may be desirable to have some or all of the users of the online system learn to identify themselves to the system. This can be done on a voluntary basis, or it can be enforced on the basis of a user group, by physical terminal, or by the entire system.

When terminal operators use the /SIGN ON command, they supply a user ID and password and, optionally, a group name. IMS passes this information to RACF. If a profile is found for that user, RACF checks the supplied password against the password in the user's profile. Before the comparison, both the password in the user profile and the supplied password are encoded using RACF's implementation of the DES algorithm. If the supplied password is correct, RACF then checks to see if the user's password has expired. If it has, RACF returns a code indicating this to IMS. IMS returns a message to the user indicating that a new password is required.

If the password is correct and has not expired, RACF checks to see if the user is a valid member of the group to which the user is signing on (if a GROUP operand was supplied), or else uses the default group name from the user profile. Additional checks are made concerning group membership.

Any time an IMS terminal is signed on, a type X'16' log record is written. This log record contains the physical terminal identifier, the user identifier, and the IMS time-stamp. A X'16' record is also written when the terminal is signed off, either as the result of a /SIGN OFF command or by another /SIGN ON command being entered from that same terminal.

If a terminal is signed on, the user identifier from the RACF token is placed in several other log records, including:

- The input message log record (X'01')
- The output message log record (X'03')
- All database change records (X'50', X'51', and X'52')

In addition, the user identifier is made available to the application program in an extension of the IO/PCB.

These facilities enhance the ability to use the IMS log records as an audit trail, whether or not RACF is used to control access to any IMS resource.

To cause IMS to use RACF for a sign on, the only thing you need to do is an IMS system generation with two operands in the SECURITY macro:

```
SECURITY TYPE=(,RACFTERM)
         ,SECLVL=(,SIGNON)  (or 'FORCSIGN')
```

The RACFTERM operand specifies that IMS is to invoke RACF to identify and verify the user who issues the /SIGN ON command.

SIGNON or FORCSIGN provide the appropriate values for the EXECUTE statement in the IMS procedure. SIGNON can be overridden by the master terminal operator during /NRE processing to deactivate sign on processing. FORCSIGN indicates that the master terminal operator cannot override sign on processing during an IMS start.

If this is all that is done, terminal operators can optionally issue the /SIGN ON command. If operators have been defined to RACF, and if they provide their

password correctly, the /SIGN ON command is accepted and all of the actions described earlier take place. (A rejected sign on results in a DFS2467 message and a X'10' log record is written.)

To define a user to RACF, a person with proper authorization issues the following RACF command from TSO:

```
ADDUSER userid NAME(user-name)
```

This RACF command causes a user profile to be created in the RACF data base with a number of defaults. The user's password is set equal to the definer's group name and is marked "expired". The user's default group is set to the definer's logon group. The defined user has no unusual authorities of any kind as far as RACF is concerned. (To find information on the ADDUSER command, see *z/OS SecureWay Security Server RACF Command Language Reference* or enter HELP ADDUSER from a TSO terminal.)

You can make a user a member of more than one group by using the RACF CONNECT command.

Without doing anything other than just described, a user can sign on to IMS, but is not required to. (Note that any RACF-defined user can use the /SIGN ON command to gain access to a terminal connected to an IMS control region.)

There are several options available to ensure that users sign on to a particular IMS control region. These options are described from the simplest to the more complex.

The easiest method to enforce sign on for all users is to run the IMS security maintenance utility (SMU) with the following:

```
)(SIGN
  STERM ALL
```

Assuming that the IMS system generation options referenced previously had been done, this run of the security maintenance utility and the EXECUTE options specified by the SECLVL operand indicate that IMS is to pass the terminal identifier of the terminal to RACF when a user signs on. If the RACF TERMINAL class is active, RACF checks the user's authority to use the terminal.

The return code RACF supplies to IMS indicates whether access to the terminal is to be allowed. There are conditions in RACF that determine this return code. The first of these conditions is set by the RACF SETROPTS command:

```
SETROPTS CLASSACT(TERMINAL) TERMINAL(READ)
```

This command sets the TERMINAL class of resource in RACF to an active, system-wide status. All subsystems that use RACF to control access to terminals now have terminal checking active when this command is issued. The READ option of the TERMINAL operand indicates how RACF is to view terminals that are not defined to RACF. READ indicates that, if RACF cannot find a profile for that terminal, access to the terminal is to be allowed.

The security maintenance utility STERM ALL operands tell IMS that the transactions can be entered only from signed-on terminals. (With READ specified for undefined terminals, the IMS terminals do not have to be defined to RACF, but all terminal operators who use that IMS control region must identify themselves through the use of the /SIGN ON command.)

If you want to be selective about which users are to be forced to sign on, see “Controlling Access to IMS Physical Terminals” on page 433.

Note that the IMS command processor is not conversational. Therefore, there is no opportunity to format the screen before a sign on. Because the screens on the terminals are not formatted for the `/SIGN ON` command, the user’s passwords are displayed. IBM recommends that you create message format screens to provide the sign on function so that the operator can enter `/FOR SIGN` and receive a formatted screen that requests the password in a non-display field.

Controlling Access to IMS Control Regions

With the steps covered in the previous topics in place, a user must use the `/SIGN ON` command to enter a transaction from an IMS terminal. The only problem here is that any RACF-defined user could use the `/SIGN ON` command to gain access to the IMS control region.

To restrict access to a control region to just those users of the system who are authorized by the nature of their jobs to access it, you must activate the APPL class in RACF as follows:

```
RDEFINE APPL (ims-id1,ims-id2,,,,,,ims-idn) UACC(NONE)

PERMIT ims-id1 CLASS(APPL) ID(groupname-1,,,,groupname-n) ACCESS(READ)

SETROPTS CLASSACT(APPL)
```

where *ims-idn* is the IMSID value specified on the IMSCTRL macro. (The default value for IMSID is `IMSA`.)

IMS supplies the control region’s application identifier to RACF during sign on. RACF checks to see if the user who is signing on has permission to use the resource pointed to by the APPLID. If no profile in the APPL class has been defined to RACF, it is assumed that the application is unprotected. RACF makes this check only at the time a terminal operator uses the `/SIGN ON` command.

When IMS requires sign on, all users of a particular IMS control region must be identified and their user IDs, or the names of the groups to which they belong, must be in the access list for that particular control region name.

Note: You should consider assigning user IDs with the PROTECTED attribute to the started procedures associated with IMS control regions. For more information, see “Using Started Procedures” on page 143.

Controlling Access to IMS Transactions

The TRANAUTH value that you specify with the SECLVL operand of the SECURITY macro causes IMS to use RACF to control access to IMS transactions:

```
SECURITY TYPE=(,RACFTERM)
         ,SECLVL=(TRANAUTH,SIGNON)
```

You can also include the TRANAUTH value in the initial system generation described earlier in this chapter. As long as the RCLASS value agrees with the RACF class descriptor table (CDT), the fact that no transaction profiles have been defined does not affect IMS operation.

The TRANAUTH value tells IMS to do two things:

- During initialization of the control region, IMS issues RACROUTE REQUEST=LIST,GLOBAL=NO to copy all of the profiles in the specified class (TIMS, GIMS) into system storage.
- When IMS receives a message, IMS issues REQUEST=FASTAUTH to check the user's authority to the resource. REQUEST=FASTAUTH checks the in-storage profiles to determine the user's authority. If the profile is found, REQUEST=FASTAUTH checks to see whether the user is allowed to access the resource. The return code from REQUEST=FASTAUTH indicates the status of the request: access-allowed, access-not-allowed, or profile-not-found. Because IMS treats the profile-not-found return as a not-protected condition, it is perfectly acceptable to have transaction authorization active, but not have any profiles specified for a given control region. The amount of overhead caused by a not-found condition is small.

The RACFTERM value tells IMS to use RACF to process the /SIGN ON command.

Grouping IMS Transactions under a Common Profile

Because of the way IMS uses RACF, users are not the only entities that can be grouped under a single name. IMS transactions can also be grouped under a common profile. Assuming that the order entry process requires the use of four different IMS transactions, they could all be defined to RACF at the same time and all could share a common access list:

```
RDEFINE Gxxx ORDENT ADDMEM(TRAN1,TRAN2,TRAN3,TRAN4) UACC(NONE)

PERMIT ORDENT CLASS(Gxxx) ID(GROUP1) ACCESS(READ)
```

Issuing these two RACF commands is the same as issuing RDEFINE and PERMIT commands for each of the four transactions, but there is no need to define the transactions individually. They share a common profile containing all other RACF information such as ownership, UACC, statistics, and the access list.

You can, of course, have an individual profile for a transaction that is already defined in the G*xxx* class. If both a group profile and an individual profile exist, REQUEST=LIST merges the two profiles and uses the merged profile. If there are conflicting specifications in the two profiles, REQUEST=LIST resolves these through a set of rules that can be specified by flags in RACF exit routines. Although RACF allows you to name a transaction more than once (for example, once as a member of an entity in the GIMS class and once as a member of the TIMS class), you should avoid it, because the REQUEST=LIST function merges the access lists from the two entries. This merging usually causes excessive use of storage and access lists that are not what the administrator intended.

After you have set the IMS classes active with the RACF SETROPTS command, and the IMS start up procedure specifies TRANAUTH, the transactions defined to RACF at that time are RACF-protected. IMS issues REQUEST=FASTAUTH at four different times:

- Before placing a transaction in the scheduler message block
- When a CHNG call is issued to a modifiable IO/PCB
- When an ISRT call to a scratch pad area contains a transaction name
- When the /SET, /LOCK, and /UNLOCK commands contain transaction names

Note: Checking calls have an effect on a transaction-to-transaction switch. A terminal operator could sign on and enter a transaction that may or may not be RACF-protected. If the transaction were to stay in the queue long enough to be scheduled after the operator has signed off, there would no longer be a

valid RACF token associated with the terminal. If the first transaction then invoked a protected transaction through a CHNG call to a modifiable IO/PCB, the scheduling of the protected transaction could fail.

IMS uses RACF to force reverification that the operator who signed on to a given terminal is the same one who is entering a second or subsequent transaction. This is done by specifying 'REVERIFY' in the APPLDATA field of the transaction profile:

```
RDEFINE Txxx tran-name UACC(NONE) APPLDATA('REVERIFY')
```

Each time users enter this transaction code, they must enter their RACF password in the place where an IMS password would go if the transaction were password-protected.

Using this method prevents the problems associated with terminal operators leaving their terminals in a signed-on state. However, it has adverse implications on message formats and on usability and productivity. It would be better if supervisors impressed on the terminal operators the importance of not leaving their terminals in a signed-on state.

The transaction authorization exit in IMS could be used to check the elapsed time between transactions, but even this does not ensure that a transaction cannot be invoked by a person other than the one who signed on and left the terminal unattended.

You can use generic names in the transaction class (T*xxx*) or as members in the group class (G*xxx*) to cover multiple transactions with similar names and authority requirements.

Controlling Access to IMS Physical Terminals

If you want to treat IMS terminals as resources (that is, to allow only certain users or groups of users to use physical terminals that are connected to IMS), you must describe the terminals to RACF and build access lists for them.

If a terminal is defined to RACF and the resource manager supplies the terminal identifier to RACF during RACROUTE REQUEST=VERIFY processing, RACF returns a code that indicates whether this user is allowed to use this particular terminal. As was indicated earlier, the signal to IMS to supply the terminal identifier comes from the security maintenance utility.

```
)(SIGN
  STERM nodename     (VTAM,TCAM)
  STERM lll#ttt       (BTAM relative line and term #)
```

Entries like the above cause IMS to pass the terminal identifier to RACF during /SIGN ON processing. If the TERMINAL class is active, RACF performs terminal authorization checking to ensure that the user is authorized to sign on to that terminal. For a complete description of setting protection for terminals, see “Protecting Terminals” on page 235.

Authorization to IMS/ESA Control Region Resources

IMS control region resources, such as program specification blocks, transaction names, and logical terminal names, are accessible to programs operating in dependent regions. To MVS, these dependent regions are normal MVS jobs, and they can be initiated through the MVS job entry subsystem by anyone. Thus, a person who is not authorized to access a database through a RACF-protected IMS

transaction could access it through a batch message processing region by putting the proper values in the EXECUTE statement for the BMP.

The application authorization security or resource access security facility of IMS can use RACF to provide additional control of who can access the resources of the control region through dependent regions. The vehicle through which this is done in IMS is the application group name (AGN).

Defining Application Group Names for IMS

Application group names are defined to IMS through the security maintenance utility. They can be used with or without RACF. If RACF is not used, you must use an installation exit routine to authorize access to control region resources by dependent regions. The supplied installation exit routine denies all access with resource access control active.

When IMS resource access security is implemented using RACF, both IMS and RACF must be aware of the resource names. You must define profiles in a RACF general resource class named A*xxx*, where A*xxx* is derived from the RCLASS=*xxx* value of the IMSCTRL macro. The default value is AIMS. IMS does not use this facility until told to do so through the system generation process and a start of IMS.

To define the application group names (AGNs) to RACF, issue the RACF RDEFINE, PERMIT, and SETROPTS commands as with other resource classes:

```
RDEFINE Axxx (list-of-agnames) OWNER(IMSADMIN) UACC(NONE)

PERMIT agname-1 CLASS(Axxx) ID(appropriate-users-or-groups) ACCESS(READ)
PERMIT agname-2 CLASS(Axxx) ID(appropriate-users-or-groups) ACCESS(READ)

SETROPTS CLASSACT(Axxx)
```

Requesting Authorization Checking

To cause IMS to request authorization checking during dependent region initialization, you must code the RACFAGN operand in the SECURITY macro for the IMS system generation. You code it in the TYPE=(RACFAGN,,) statement.

There is no capability to allow the master terminal operator to override this option (as there is, for example, in other options in the SECLVL statement.) The RACFAGN statement causes the inclusion of the call to RACF and conditions the job control language for the control region to have the value ISIS=1 in the EXECUTE statement for it. The ISIS=1 specification indicates that RACF is to be used for resource access security. ISIS=2 says use the resource access exit routine written by the installation. ISIS=0 says do not use resource access security.

Again, the master terminal operator cannot override these values during a restart of the system. However, they can be affected by the RGSUF option in the IMS EXECUTE statement.

Note: Before activating resource access security with RACF, it is important to note that all dependent region job control language is affected by this feature. If you start IMS control regions as batch jobs, all job statements must contain valid RACF-defined user IDs.If you use surrogate or propagated batch job submission, you can define these user IDs as protected user IDs (see “Using Protected User IDs for Batch Jobs” on page 444), or you must supply current passwords on all job statements. If you choose to supply passwords, you may define these users to RACF with passwords that never expire. For this reason, IMS supplies a DDNAME, IMSJOBS, whose data set should be

RACF-protected with a UACC of NONE, and the user ID associated with the IMS started procedure should be in the access list with at least READ permission.

In addition to this, the EXECUTE statements for the dependent regions must specify the IMSID of the control region (IMSCTRL macro), the application group name (defined to RACF and in the security matrix), and, for batch message processing regions, the names of the resources to be accessed by this region.

You must run the security maintenance utility to assign control region resource names to a given application group name.

For message processing regions, the EXECUTE operands include only the control region name and the application group name. For example:
If a message processing region tries to schedule a transaction that is not in the

```
)(AGN  agname-2
   AGTRAN ALL
   AGLTERM lterm-n
```

Figure 49. Message Processing (MPP) Example

application group name for that region, the scheduling fails.

For batch message processing regions, the EXECUTE operands include the control region name, the application group name, and the names of the transaction, the program specification block, and the logical terminal that the program wishes to use. For example:

```
)(AGN  agname-1
   AGPSB psb-1
   AGPSB psb-2
   AGPSB psb-n
   AGTRAN tran-1
   AGTRAN tran-n
   AGLTERM lterm-1
   AGLTERM lterm-n
```

Figure 50. Batch Message Processing (BMP) Example

The Checking Process

The checking process is a two-step process. The first check involves RACF. The second does not.

1. Each dependent region running as a batch job has a user ID associated with it. During dependent region initialization, IMS performs a RACF authorization check to see whether the user can use the application group name. IMS performs this check using the RACF class name (A*xxx*) and the name of the AGN passed to it in the EXEC statement parameter list. If the user is not allowed to use the application group name, RACF returns a “not authorized” return code, and IMS does not allow the dependent region to connect. If RACF returns an “authorized” return code, the connection is made.

   For the dependent region to make the connection to the control region, the JOB statements that start the dependent region must have a valid user ID and a current password for the user. To prevent unauthorized disclosure of the passwords in these JOB statements, an IMS system data set has been included in the procedures that bring up the IMS control region. This data set is

represented by the IMSJOBS DD name. It should be RACF-protected, and the IMS started procedure name should be in the access list with READ access. No one should have access to this data set as a matter of course. When the data set must be changed, RACF commands can be used to allow update access to it. The access authorization can then be revoked after the data set has been updated.

2. The second part of the two-step process is an IMS function only. IMS checks the name of the transaction or PSB or logical terminal that is being requested by the dependent region against the entries in the security matrix and allows or disallows use depending on whether the name is in the entry for the application group name.

   For message processing regions, application resource security is somewhat like class scheduling in that transactions can be scheduled only in regions whose application group name allows them.

   For batch message processing regions, this level of control prevents unauthorized users from starting an MVS job that can access the resources defined in the control region.

Summary

You can enhance the security and integrity features of IMS to a significant degree by using RACF.

Any security mechanism is only as good as the management control of the people who use the system. IMS and RACF provide the tools to enhance control of a critical resource. It is management's responsibility to see that the controls that are implemented are working the way they are supposed to work, and that variances are reported to and acted on by management.

In this regard, RACF, with its lists of users and lists of resources, allows management to delegate the authority to the owners of these entities in such a way as to maintain the separation of duties while maintaining a flexible, responsive access control strategy.

In order to be effective, access control must allow management to adopt the principle of "least possible privilege" for those resources that are deemed to be highly sensitive. This principle says that access to these resources is controlled in such a way that permission to use them is restricted to just those people whose normal duties require their use. Any unusual use of the resource should be approved by an administrator or manager, as well as the owner of the resource.

The delegation mechanism in RACF and the easy, nontechnical commands that change the relationship of a user to a resource mean that adopting the principle of "least possible privilege" need not be burdensome nor inflexible when unusual circumstances dictate that access permission should be changed. When an unforeseen circumstance requires a change in access privilege, the change can be made by a nontechnical person with access to a TSO terminal, and management can be alerted to review the fact that the change was made.

Through the use of RACF, the security functions of IMS can move out of the highly centralized environment required previously and into a more flexible, responsive, and secure environment.

Chapter 15. Providing Security for JES

You can use JES initialization statements, JES installation exits, RACF, or any other functionally equivalent program to provide security for JES.

This chapter describes how to use RACF to provide security for JES. In this chapter, the term JES refers to both JES2 and JES3, except when there is a difference between the two.

Planning for Security

You and your JES system programmer should develop a security plan for JES. Together, you should determine which resources you want to protect and decide who should have access to those resources. Your security plan should address questions such as:

- What resources must I protect?
- Should I restrict jobs and users from certain information depending on criteria such as security labels?
- Should I limit the job names users can submit or cancel?
- Is it important to protect SYSIN and SYSOUT?
- Which remote workstations should access my system?
- Can other nodes submit jobs to my system?
- To which nodes should I allow my system to send data?
- Should I limit the commands an operator can use?
- Do I want to restrict the consoles an operator can use to enter certain commands?
- What commands should I allow jobs, workstations, and nodes to submit to my system?
- Do I want only selected output devices to process particular output?
- Should the security label of the output appear on the header pages?

You must gather a great deal of information from your JES system programmer about specific resources and access requirements. If JES2 is installed on your system, the JES system programmer should use *z/OS JES2 Initialization and Tuning Guide*.

If JES3 is installed on your system, the JES system programmer should use *z/OS JES3 Initialization and Tuning Guide*.

How JES and RACF Work Together

JES requests RACF services by issuing the RACROUTE macro. The MVS system authorization facility (SAF) handles the RACROUTE macro invocation. If RACF is installed, SAF passes the security information specified by JES on the RACROUTE macro invocation to RACF. RACF evaluates the security information and returns the results of that evaluation to JES. JES then enforces the results of the security check, such as permitting or denying access to a data set, or allowing a job to execute.

Your JES system programmer can use JES2 macros to place additional calls to SAF in JES installation exits. See *z/OS JES3 Customization* or *z/OS JES2 Macros* and *z/OS JES2 Installation Exits*.

JES Code That Can Bypass RACF Protection

Your installation can use JES initialization statements and JES installation exits to provide protection and, in some cases, to bypass RACF protection. For more information, see *z/OS JES2 Initialization and Tuning Guide* or *z/OS JES3 Initialization and Tuning Guide*.

Defining JES as a RACF Started Procedure

For performance and ease of migration, JES should be defined as a started procedure *with the trusted attribute* when RACF is installed.

To do this, take the following steps:

1. Ask your JES system programmer for the name of your JES started procedure.
2. Verify that the name of your JES started procedure exists as one of the following:
   - An entry in the STARTED class. If it is not defined, you need to create one. For example, if the name of your started procedure is JES2, issue:

     ```
     SETROPTS GENERIC(STARTED)
     RDEFINE STARTED JES2.* STDATA(USER(jes-userid) TRUSTED(YES))
     ```

     Be sure to refresh the class after you create the entry. Issue:

     ```
     SETROPTS RACLIST STARTED REFRESH
     ```

     Note: This is the preferred method.
   - An entry in the RACF started procedures table (module ICHRIN03). If none is defined, create an entry for JES.

   In either case, be sure that JES has the *trusted* attribute.
3. Create a user profile for the JES started procedure:

   ```
   ADDUSER jes-userid
           DATA('JES started procedure')
           PASSWORD(password)
           DFLTGRP(appropriate-group)
   ```

For more information on adding a started procedure, see “Using Started Procedures” on page 143 and *z/OS SecureWay Security Server RACF System Programmer’s Guide*.

Forcing Batch Users to Identify Themselves to RACF

To prevent unauthorized users from running batch jobs, you can require all batch jobs to have RACF identification. To do this, enter the following:

```
SETROPTS JES(BATCHALLRACF)
```

When you specify BATCHALLRACF, any batch job that does not have a RACF-defined user specified on the USER parameter of the JOB statement, or propagated security information associated with it, fails.

Specifying NOBATCHALLRACF allows such jobs to run.

Support for Execution Batch Monitor (XBM) (JES2 Only)

Execution batch monitor (XBM) jobs are processed in the same way as normal batch jobs. To require all XBM jobs to have RACF identification, enter:

```
SETROPTS JES(XBMALLRACF)
```

With this operand in effect, any XBM job that does not have a RACF-defined user ID and password on the JOB statement, or propagated RACF identification associated with it, fails.

Specifying NOXBMALLRACF allows XBM jobs to run without RACF user IDs.

Note: On systems with the XBMALLRACF operand in effect, the BATCHALLRACF operand controls batch jobs *other than* jobs that run under an execution batch monitor (XBM).

Defining and Grouping Operators

Your security plan may require that system programmers and operators at your installation be defined to RACF. To improve accountability, you should create user profiles for any persons who:

- Issue JES commands
- Enter commands from an MCS-managed console
- Update system data sets that JES uses

You can organize your installation's support personnel, particularly operators, into groups that are responsible for a particular area. For example, you may want to group your installation's support personnel by shift, functional area, or both.

To identify and group your installation's support personnel, you should:

- List the user IDs of all of the system programmers and operators at your installation
- Group any of the user IDs together if they:
  - Perform similar tasks
  - Work on the same shift
  - Are responsible for the same area

  If you decide to combine users into groups, you must:
  - Record all of the user IDs in the group
  - Describe why the users were grouped together
  - Assign a unique name to the group

You should keep a record of user IDs and group name handy for use in securing other system resources such as spool data, console access, and commands as well as for updating groups in the future.

JES User ID Early Verification

Early verification is always done, even if the SETROPTS command has been issued with JES(NOEARLYVERIFY) specified.

User ID Propagation When Jobs Are Submitted

For each previously-validated RACF user who submits a batch job to JES through a JES internal reader, SAF propagates the following security information to the batch job:

- If USER is not specified on the JOB statement, the current RACF user ID is used.
- If PASSWORD is not specified on the JOB statement, the current user password is not required if the submitter propagates.
- If SECLABEL is not specified on the JOB statement, the submitter's current security label is used.

Note: If GROUP is not specified on the JOB statement, the default connect group is used from the user profile of the user used for the job.

This has the following advantages:

- It reduces the possible exposure of security information (especially passwords) stored in clear text in JCL.
- It reduces administrative overhead of maintaining RACF user IDs, passwords, and security labels in the JOB statements for all batch jobs.

As a result, a TSO user, for example, is not required to specify this security information for each job submitted.

Note: You can prevent user ID propagation for specific users. See "Controlling User ID Propagation in a Local Environment" on page 443.

Allowing Surrogate Job Submission

You can allow the use of surrogate users. A surrogate user is a RACF-defined user who has been authorized to submit jobs on behalf of another user (the execution user) without specifying the execution user's password. Jobs submitted by the surrogate user run with the identity of the execution user. For example, if user JOE submits a job with the following JOB statement, JOE is the surrogate user and TOM is the execution user:

```
//jobname JOB   'accounting-information',USER=TOM
```

All access checks are done with TOM's user ID. Any auditing records produced by RACF because of the surrogate job's actions include the information that the job is a surrogate job (unless the PASSWORD parameter is specified on the JOB statement).

Attention

A user should not allow another user to act as surrogate user *unless the surrogate user can be trusted as much as the execution user is trusted.* This is because the surrogate user can do anything the execution user can do (unless the surrogate user lacks access to a security label that protects a resource). For example, the surrogate user can submit a job to copy, alter, or delete the execution user's data.

The surrogate user must specify the execution user's user ID on the USER parameter on the JOB statement and must *not* specify a password. If the PASSWORD parameter is specified with a password, surrogate processing is not performed, and audit records generated by the job's activities do not indicate that the job is a surrogate job. This applies not only to jobs submitted through the TSO SUBMIT command, but any time the submitter is a RACF-defined user.

Note: If the SECLABEL class is active, and the SETROPTS MLS option is in effect, the security label that the job runs under must be equal to or greater than the current security label of the submitter, and the submitter must have READ access authority to the security label under which the job runs. After job verification is complete, a job submitted by a surrogate user runs as though the execution user had submitted the job. If a SECLABEL is not specified, the surrogate user's job gets the surrogate's default SECLABEL, *not* the default SECLABEL of the execution user.

To allow surrogate users, do the following:

1. Ensure that the installation exit for the TSO SUBMIT command (IKJEFF10) does not prevent users from submitting jobs with job names that do not match their user IDs. The installation exit supplied by IBM meets this requirement, because it does not check the JCL of submitted jobs. For more information, see *z/OS TSO/E Customization*.
2. If your installation implemented the sample ICHRTX00 exit from SYS1.SAMPLIB member RACINSTL to enable surrogate user processing, you should migrate to profiles in the SURROGAT class. After RACF is installed, the code in the ICHRTX00 exit that checks $SUBMIT.userid profiles is not used. You should copy the $SUBMIT.*userid* profiles to SURROGAT profiles as follows:

   ```
   RDEFINE SURROGAT execution-userid.SUBMIT
           FROM($SUBMIT.execution-userid) FCLASS(FACILITY)
   ```

3. Define resource profiles in the SURROGAT class for each execution user who needs to allow others to be surrogate users:

   ```
   RDEFINE SURROGAT execution-userid.SUBMIT UACC(NONE) OWNER(execution-userid)
   ```

   Note: Specifying the OWNER operand allows the execution user to issue the PERMIT command for this profile.

4. To specify that another user can act as the surrogate for an execution user, give the surrogate user READ access authority:

   ```
   PERMIT execution-userid.SUBMIT CLASS(SURROGAT) ID(surrogate-userid) ACCESS(READ)
   ```

   Only users and groups that have READ access authority are allowed to submit jobs on behalf of another user.

To check whether a user can submit jobs for another user, make sure the user (or a group the user is a member of) is in the access list with READ access authority:

```
RLIST SURROGAT execution-userid.SUBMIT AUTHUSER
```

5. When you are ready to control access using the SURROGAT profiles, activate the SURROGAT class:

   ```
   SETROPTS CLASSACT(SURROGAT)
   ```

   To “turn off” surrogate support for a particular user, delete the profile for that user. To “turn off” surrogate support for all users, deactivate the SURROGAT class.

NJE Notes: The node in which SURROGAT checking occurs depends on the job statements (see “Where NJE Jobs Are Verified” on page 444). For verification done on the receiving node, the SURROGAT checking is done *after* any translation due to NODES profiles (see “Understanding NODES Profiles” on page 453).

If the submitter of a job is a started procedure, the execution node is not checked during SURROGAT processing.

Controlling User ID Propagation in a Local Environment

In some environments, such as CICS, jobs submitted without the USER operand specified on the JOB statement run under a user ID other than the user submitting the job. For example, if a user running under CICS submits a batch job without specifying a user ID on the JOB statement, the job runs under the CICS user ID and has the access authorities of the CICS user ID.

You can prevent the CICS user ID from being propagated to these batch jobs by defining a profile whose name is the CICS user ID. Follow these steps:

1. Define a profile in the PROPCNTL class where the profile name is the user ID that is not to be propagated (in this case, user ID CICS1 is not to be propagated):

   ```
   RDEFINE PROPCNTL CICS1
   ```

2. Activate the PROPCNTL class:

   ```
   SETROPTS CLASSACT(PROPCNTL)
   ```

3. Issue the SETROPTS command with the RACLIST operand to activate SETROPTS RACLIST processing for the PROPCNTL class:

   ```
   SETROPTS RACLIST(PROPCNTL)
   ```

   If you do not activate SETROPTS RACLIST processing for the PROPCNTL class, RACF ignores the profiles you create in this class.

The above sequence of commands eliminates user ID propagation of the user ID CICS1.

Notes:

1. For a profile in the PROPCNTL class, RACF checks only for the presence or absence of a profile in this class. If a profile exists for a particular user ID, user ID propagation does not occur for that user ID.
2. RACF performs no logging and issues no messages for profiles in the PROPCNTL class.
3. PROPCNTL profiles only control propagation for local jobs. If the installation uses &RACLNDE for its remote nodes, only the PROPCNTL profiles are

necessary. For more information on the use of &RACLNDE, see “Defining Nodes as Local Input Sources” on page 469. If the installation uses NODES profiles, it must also use them to control propagation (see “Controlling User ID Propagation in an NJE Environment” on page 460).

4. If you have controlled a user ID using the PROPCNTL class, and that user wants to submit a batch job to run from that user ID, the JOB statement must contain both the user ID and proper password. For example, if user A submits a job with USER=A, PASSWORD=password must also be specified.

   However, if a different user wants to submit a job using the controlled user ID, that user may either specify the user ID and password as above, or may use the facilities provided by the SURROGAT class and just specify the user ID. For example, if you controlled user A using the PROPCNTL class, user B could submit a job, specifying only USER=A with the appropriate SURROGAT authorization.

Using Protected User IDs for Batch Jobs

You can define the user IDs associated with batch jobs as *protected* user IDs. This will prevent them from being revoked through inadvertent or malicious incorrect password attempts, or from being used for another purpose where a password is normally supplied, such as logging on to the system. See “Defining Protected User IDs” on page 86 for information on implementing protected user IDs.

In order to execute a batch job using a protected user ID, you must submit the job through a means that does not require a password, such as through user ID propagation or surrogate job submission. Jobs that are submitted with `USER=` and `PASSWORD=` specified on the JOB statement can not be associated with protected user IDs.

Propagating Protected User IDs

If a started procedure or job executes with a protected user ID (for example, USERA) and user ID propagation is enabled for USERA, any job submitted by USERA that does not have `USER=` specified on the job statement will execute with a protected user ID, USERA.

See “Assigning RACF User IDs to Started Procedures” on page 143 for information on using protected user IDs for started procedures.

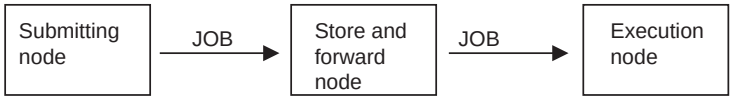
Using Protected User IDs for Surrogate Job Submission

Surrogate user IDs and execution user IDs can be defined as protected user IDs. A started procedure or job that executes with a protected user ID can be authorized to submit jobs that have `USER=` specified on the job statements. The execution user IDs associated with the submitted jobs may also be defined as protected at your option.

A started procedure or job that does not execute with a protected user ID can be authorized to submit jobs that execute with protected user IDs.

Where NJE Jobs Are Verified

The following is a simple network showing the path of a job.



User verification for NJE jobs normally is done at the execution node. However, RACF authorization checking may occur additionally at the submitting node, depending on the following:

- Those jobs sent using the JES2 /*ROUTE XEQ statement or /*XEQ statement are verified at the execution node only.
- Those jobs sent using the JES2 /*XMIT statement or the JES3 //*ROUTE XEQ or //XMIT statement have their first JOB statement verified at the submitting node and their second JOB statement verified at the execution node.

Submitter information is propagated from trusted nodes. The submitter information is:

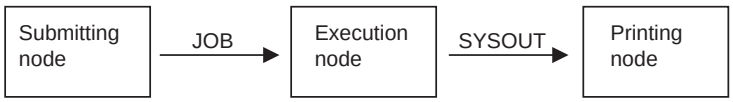
- The token for a verified first job card
- The original submitter's token if the job was submitted from an internal reader
- The "unknown user" token if the job was submitted from a physical reader
- NJE header information (no token available) if the job was submitted from a downlevel node

Whether a job is accepted is based on a combination of the submitter's ID, group, or security label. Whether security information is propagated and translated is based on the submitter's ID (as taken from above). Job acceptance and translation is done using profiles in the NODES class. RACF finds the best fit among the profiles in the NODES class and uses the information specified in the UACC and ADDMEM information.

For more information, refer to "Authorizing Network Jobs and SYSOUT (NJE)" on page 452.

How SYSOUT Requests Are Verified

The following is a simple network showing the path of a job:



For inbound SYSOUT, user verification occurs at the printing node instead of the submitting node (as it can for inbound jobs). On the printing node, RACF authorization checking occurs in the NODES class, as it does for inbound jobs. RACF finds the best fit among the profiles in the NODES class and uses the information specified in the UACC and ADDMEM information.

Whether the SYSOUT is accepted is based on a combination of the owner's ID, group, or security label. Whether the security information is accepted and translated is based on the owner's ID taken from:

- The job token from the NJE header as verified at the executing node
- If no token is available (SYSOUT is from a downlevel node), the owner is considered to be the NJE undefined user as defined by:

  `SETROPTS(JES(NJEUSERID(`*`userid`*`)))`

In addition, if &SUSER (submitting user) is specified on the ADDMEM operand, the submitter can be used as the owner if one of the following is also true:

- The submitting node is defined as a local node in the &RACLNDE profile in the RACFVARS class. In this case, the submitting user and group are used as the SYSOUT owner values and are unchanged (no translation).
- The NODES profile that matches is the profile named *submitting-node*.USERS.*submitter* and UACC(CONTROL) is specified.

  If there is a translate value, but it is not &SUSER, the SYSOUT owner user ID is the translate value. If it is &SUSER, the owner is the unchanged submitter user ID. In addition, a look-up is done for the NODES profile that matches the form *submit-node.GROUPS.submit-group*. If this profile has an ADDMEM translate value, that value is used as the SYSOUT owner group. Otherwise, the unchanged submit group is used. The UACC for this profile does not matter.

Security Labels for JES Resources

When your installation uses security labels, each protected JES resource can have a security label associated with it. For spooled data sets, JES maintains the security label with the data set itself (not in a RACF profile). For other resources like consoles and DASD data sets, RACF maintains the security label in the resource profile that protects the resource.

Attention

Security labels should be consistent throughout a JES complex to prevent information from being declassified. To check what your system configuration should be, see “B1 Configuration Requirements” on page 12.

For more information about security labels, see “B1 Security Level” on page 11.

Controlling Access to Data Sets JES Uses

The JES spool and checkpoint data sets are critical for proper operation of your JES system. It is critical that JES be the only user that can update the information in these data sets. However, a limited group of users must be able to re-create the spool and checkpoint data sets (should the data set become unusable because of hardware problems). Also, restrict access to the data set that contains the modules that JES uses. Make sure profiles exist for any data sets you might use for JES2 checkpoint reconfiguration.

You can define data set profiles to protect the system data sets that JES uses to control its own processing. Protecting your installation’s system data sets prevents unauthorized users or jobs from accessing, modifying, or destroying critical system data.

The JES system programmer should supply you with the following information for each data set to be protected:

- The name of the data set
- The universal access authority to be associated with the data set
- The security label to be associated with the data set (if labels are being used)
- Whether audit records should be generated:
  - Each time the data is accessed
  - When an unauthorized attempt is made to access the data set

– When an authorized attempt is made to access the data set

Make sure to define JES as a started procedure with the trusted attribute. See “Defining JES as a RACF Started Procedure” on page 439.

Controlling Input to Your System

You can use RACF to ensure that jobs entering your JES system are authorized for processing. JES carries security information about the submitter of a job internally and invokes the services of RACF at key points during JES processing, such as when a job enters the system through a TSO SUBMIT command or through a device reader.

You can protect several job entry resources including the use of job names and sources of job entry such as internal readers, device readers, RJP, RJE, and network nodes. This section describes how RACF validates users, how to control the use of job names, and how to control the use of input sources. “Authorizing Network Jobs and SYSOUT (NJE)” on page 452 provides a separate discussion of how to control security for inbound and outbound network jobs and SYSOUT.

How RACF Validates Users

When RACF is active, RACF ensures that the job’s password, user ID, group name, and security label are valid before allowing the job to be processed. If security labels are being used, JES obtains the label from the job card. If the job card does not specify a label, RACF obtains the security label from the RACF profile associated with that job’s user ID. If no security label exists in the RACF profile, the job is automatically assigned a security label of SYSLOW.

The extent to which RACF performs user validation for jobs entering the system through NJE nodes depends on the universal access authority assigned to that node. “Authorizing Network Jobs and SYSOUT (NJE)” on page 452 lists those values and their effects on user validation.

You can allow the setting up of surrogate users. A surrogate user is a user who submits jobs on behalf of another user.

Surrogate job submission allows a user to submit jobs on behalf of another user without having to specify the original user’s password. Jobs submitted by a surrogate user execute with the identity of the original user. Although the surrogate user does not have to provide the password of the original user, RACF ensures that the job’s security label overrides the surrogate’s security label and that the original user is authorized to use the security label associated with the job. For additional information about defining surrogate job submission, see “Surrogate Job Submission” on page 451.

Propagating Security Information

If RACF is active and a job enters the system with some or all security information missing, SAF propagates the submitter’s security information to the job. For example, if JOB1 submits JOB2, and JOB2 does not have SECLABEL= specified on the JOB JCL statement, SAF passes the value of SECLABEL= from JOB1’s JOB statement to JOB2.

Any user in a currently active session (for example, TSO or an executing batch job) can have SAF propagate the security information associated with the session by omitting the information on the JOB statement for the job. If SAF cannot determine the group or security label information from the current session, SAF uses the

default information in the RACF profile. However, SAF does not propagate security information to a job that enters the system from an RJE work station or a physical card reader.

Propagating Security Information across a Network

To allow users to submit jobs without specifying security information such as a user ID, JES propagates the submitter's security information when the information is omitted. For example, you can have users omit their password from their job statements if you do not want to send passwords through the network. Propagation also occurs when you use the /*XMIT card to transmit a job to another node in the network. In this case, SAF passes the information that appears in the first JOB statement to the JOB statement that follows the /*XMIT card.

If SAF is unable to verify the security information on the first JOB statement:

- If a TSO user submitted the job, SAF passes the TSO user's security information to the second JOB statement.
- If the job entered the system through a local card reader, SAF uses a default user ID for propagation purposes. (For information about the default user ID, see "Understanding Default User IDs" on page 467.)

"How JES Sends Security Information" on page 468 describes what security information RACF propagates for NJE jobs.

Controlling the Use of Job Names

You can use profiles in the JESJOBS class to control which job names users can submit or cancel.

Because most installations have many jobs and users, it might not be practical to define all of the profiles needed to authorize every job name. A more reasonable approach would be to restrict the use of only certain job names.

You can specify which job names can be entered from a specific device. For more information, see "Conditional Access Lists for General Resource Profiles" on page 206 and the note in step 4 on page 449.

Notes:

1. TSO installation exit IKJEFF53 must be modified to become a dummy exit when the JESJOBS class is active. See *z/OS TSO/E Customization* for specific information.
2. At least one generic profile with a universal access of READ is required for the TSO SUBMIT command when the JESJOBS class is active. A generic profile is not required for the TSO CANCEL command. However, without a generic profile, only the owner of a job or those granted access through RACF profiles can use the CANCEL command on data sets associated with that job.
3. The JESJOBS class does not apply to jobs submitted via RJE/RJP consoles. Refer to "Remote Workstations (RJP/RJE Consoles)" on page 478.

Controlling Who Can Submit Jobs by Job Name

To control who can submit jobs by job name, take the following steps:

1. Ask your TSO system programmer to ensure that TSO installation exit IKJEFF53 checks if the JESSPOOL or JESJOBS class is active, and if either is active, returns to the caller with no action. For specific information, see *z/OS TSO/E Customization*.

   Note: SYS1.SAMPLIB contains a sample of such an exit.

2. Define at least one profile with a universal access of READ to allow users to submit jobs when the JESJOBS class is activated:

   ```
   RDEFINE JESJOBS ** UACC(READ)
   ```

   Note: This example assumes that a SETROPTS GENERIC(JESJOBS) was previously issued to turn generics on for this class and that a SETROPTS REFRESH was then done.

3. Define profiles with UACC(NONE) for the job names you want to protect.

   ```
   RDEFINE JESJOBS SUBMIT.nodename.jobname.userid UACC(NONE)
   ```

   where:

   nodename is your local node name.

   Note: It is recommended that you define a profile in the RACFVARS class named &RACLNDE, and use &RACLNDE for all profiles in the JESJOBS class.

   jobname is the name of the job specified on the JOB statement.

   userid is the user ID under which the job is to execute (either the USER operand on the JOB statement or the propagated user ID).

   For example, the following command would prevent any user from submitting jobs whose job names begin with PAYROLL.

   ```
   RDEFINE JESJOBS SUBMIT.*.PAYROLL*.* UACC(NONE)
   ```

4. To allow users to submit jobs protected by the profile, give them READ access authority:

   ```
   PERMIT SUBMIT.*.PAYROLL*.* CLASS(JESJOBS) ID(PAYGROUP) ACCESS(READ)
   ```

   Note: By denying a user sufficient access to a SUBMIT profile, you can prevent that user from submitting jobs protected by the profile *even if that user knows the password or is an authorized surrogate user*.

   For example, the following profile would prevent jobs from being submitted with USER01 as the user ID:

   ```
   RDEFINE JESJOBS SUBMIT.*.*.USER01 UACC(NONE)
   ```

   You can also provide conditional access to the job name, depending on the class and ID of the port of entry (POE) associated with the submitter of the job. The class name you would use is determined by what the submitter is. For a regular submission from a TSO logon session, the submitter's POE is a terminal ID and the class name is TERMINAL. The submitter's POE can also be a JESINPUT device when the submitter of the job is another job.

   Making use of the job name conditional on the JESINPUT device is not recommended because this is very much dependent on what type of job was submitted. If the submitting job is a local job, its JESINPUT POE would be an internal reader, a local card reader, or an RJE reader.

   However, if the submitting job is an NJE job (for example, from another JES node), its JESINPUT POE would be the node name. This uncertainty can make the use of WHEN(JESINPUT) for the JESJOBS class difficult. Therefore, you should be careful if you decide to use it.

For example, you can allow a user to submit a job only from a certain terminal ID by specifying the WHEN(TERMINAL) operand on the PERMIT command as follows:

```
PERMIT SUBMIT.*.PAYROLL*.* CLASS(JESJOBS) ID(USER01)
       ACCESS(READ) WHEN(TERMINAL(terminal-ID))
```

where *terminal-ID* is the terminal to which the submitter is logged on.

5. When you are ready to use the JESJOBS class to control who can submit jobs, activate the JESJOBS class:

   ```
   SETROPTS CLASSACT(JESJOBS)
   ```

   Note: If you activate this class and create no profiles for it, users cannot submit batch jobs.

Controlling Who Can Cancel Jobs by Job Name

Users are always authorized to cancel jobs that they have submitted. Using RACF, you can control who can use the TSO CANCEL command to cancel jobs, depending on the job names. To do this, take the following steps:

1. Ask your TSO system programmer to change TSO installation exit IKJEFF53 to become a dummy exit. For specific information, see *z/OS TSO/E Customization*.
2. Define profiles for the job names you want to protect:

   ```
   RDEFINE JESJOBS CANCEL.nodename.userid.jobname UACC(NONE)
   ```

   Note: The qualifiers for CANCEL profiles have the same meaning as for SUBMIT profiles. However, the *jobname* and *userid* qualifiers are reversed in CANCEL and SUBMIT profiles. This is because of the expected use of the profiles:

   - It is likely that many users would submit jobs having common job names, with certain exceptions. For example, the following profiles would allow many users to submit jobs whose names begin with PAYROLL, except when those jobs run with BEN's authority:

     ```
     RDEFINE JESJOBS SUBMIT.*.PAYROLL*.*   UACC(READ)
     RDEFINE JESJOBS SUBMIT.*.PAYROLL*.BEN UACC(NONE)
     ```

   - It is likely that one user would give another the authority to cancel all of the first user's jobs, with certain exceptions. For example, the following profiles would allow JOE the authority to cancel BEN's jobs, except for his PAYROLL jobs:

     ```
     RDEFINE JESJOBS CANCEL.*.BEN.* UACC(NONE)
     PERMIT  CANCEL.*.BEN.* CLASS(JESJOBS) ID(JOE) ACCESS(ALTER)
     RDEFINE JESJOBS CANCEL.*.BEN.PAYROLL* UACC(NONE)
     ```

   - These examples assume that a SETROPTS GENERIC(JESJOBS) was previously issued to turn generics on for this class and that a SETROPTS REFRESH was then done.

3. Give users the appropriate access authority:

   ```
   PERMIT CANCEL.*.*.PAYROLL* CLASS(JESJOBS) ID(PAYGROUP) ACCESS(ALTER)
   ```

   Users must have ALTER access authority to issue the CANCEL command for the job.

4. If the JESJOBS class is not already active, activate the JESJOBS class:

   ```
   SETROPTS CLASSACT(JESJOBS)
   ```

Allowing a TSO User to CANCEL All Jobs Originating from Local Nodes

To allow a TSO user to cancel all jobs that originate on nodes you treat as local nodes, do the following:

1. Define a profile named &RACLNDE in the RACFVARS class, specifying on the ADDMEM operand which nodes are to be treated as local:

   ```
   RDEFINE RACFVARS &RACLNDE UACC(NONE) ADDMEM(POKMVS1 POKMVS2)
   ```

   UACC(NONE) is recommended to protect the &RACLNDE profile itself.
2. Define a profile in the JESJOBS class as follows:

   ```
   RDEFINE JESJOBS CANCEL.&RACLNDE.*.* UACC(NONE)
   ```

   This example assumes that a SETROPTS GENERIC(classname) was previously issued to turn generics on for this class and that a SETROPTS REFRESH was then done.
3. Give the appropriate access to the TSO user:

   ```
   PERMIT CANCEL.&RACLNDE.*.* CLASS(JESJOBS) ID(USER1) ACCESS(ALTER)
   ```

   If there are any other JESJOBS resources that begin with CANCEL, you may also need to permit users appropriate access to those.
4. If you have not already done so, activate the JESJOBS and RACFVARS classes:

   ```
   SETROPTS CLASSACT(JESJOBS RACFVARS)
   ```
5. Refresh SETROPTS RACLIST processing for the RACFVARS class for the change to take effect:

   ```
   SETROPTS RACLIST(RACFVARS) REFRESH
   ```

If, later, you decide that node POKMVS2 should no longer be treated as a local node, do the following:

```
RALTER   RACFVARS &RACLNDE DELMEM(POKMVS2)
SETROPTS RACLIST(RACFVARS) REFRESH
SETROPTS GENERIC(JESJOBS)  REFRESH
```

If, later, you decide that USER2 should also be allowed to cancel local jobs, do the following:

```
PERMIT   CANCEL.&RACLNDE.*.* CLASS(JESJOBS) ID(USER2) ACCESS(ALTER)
SETROPTS GENERIC(JESJOBS) REFRESH
```

Surrogate Job Submission

You can allow the use of surrogate users. A surrogate user is a RACF-defined user who has been authorized to submit jobs on behalf of another user (the original user) without having to specify the original user's password. Jobs submitted by the surrogate user execute with the identity of the original user. This can be useful when a person is assuming a production workload for someone going on vacation or a leave of absence.

For information on setting up surrogate users, see "Allowing Surrogate Job Submission" on page 441.

Authorizing the Use of Input Sources

You can use RACF to limit which sources of input are valid for job submission, including RJP workstations, device readers, nodes, and internal readers. For example, you might want to prevent certain users from entering jobs from a particular RJP workstation.

To authorize the submission of work from specific input sources, take the following steps:

1. Ask your JES system programmer for the following information:
   - The name of the device. This is described in the section on authorizing the use of input sources in *z/OS JES2 Initialization and Tuning Guide*.
   - The user ID or group name of the users you want to authorize or restrict.
   - The universal access authority to associate with each device. Valid access authorities for input devices are:

     NONE Specifies that the input device can be used only by those users explicitly permitted through the access list.

     READ Specifies the minimum authority required to use the input source.
2. Define a profile for each input source, as follows:

   ```
   RDEFINE JESINPUT source-name UACC(NONE)
   ```
3. It is *strongly recommended* that you create a profile with a UACC of READ for all JES input sources that are otherwise not defined:

   ```
   RDEFINE JESINPUT ** UACC(READ)
   ```

   This example assumes that a SETROPTS GENERIC(JESINPUT) was previously issued to turn generics on for this class and that a SETROPTS REFRESH was then done.

   If you do not, users can access only JES input sources to which they (or their groups) are explicitly authorized.
4. For each protected input source, grant access to the users or groups who need to use it:

   ```
   PERMIT source-name CLASS(JESINPUT) ID(user-or-group) ACCESS(READ)
   ```
5. When you are ready to start using the protection provided by the profiles you have created, activate the JESINPUT class:

   ```
   SETROPTS CLASSACT(JESINPUT) REFRESH
   ```

   If you activate this class and create no profiles for it, users cannot submit batch jobs.

Authorizing Network Jobs and SYSOUT (NJE)

This section contains information about how to use RACF to ensure that all work entering or leaving your node complies with your installation's security policy. You can control:

- Jobs and data received from other nodes in a network, as long as the inbound job or data includes a standard NJE header. This includes jobs or data from an RSCS node.
- The extent of security validation performed at your node.
- Jobs and data destined for other nodes in a network.

JES does not validate work passing through your node on its way to another node in the network, but it does protect the work from unauthorized access while the work is temporarily stored on spool at your node.

To provide security for Network Job Entry (NJE), activate the NODES class (for inbound work) or the WRITER class (for outbound work), and define the profiles

needed to enforce your installation's NJE security policy. As with other RACF-protected resources, you must gather information from your JES system programmer to define profiles.

Before you request this information, you must first work with your JES system programmer to decide whether you want to protect inbound work, outbound work, or both. For inbound work, you must decide whether you want to protect jobs, SYSOUT, or both. You must also determine which users and groups of users may submit work, and the security labels that are valid for processing on your node.

Authorizing Inbound Work

The following list outlines the topics discussed in this section:

- "Understanding NODES Profiles" explains how the values you select for the NODES profile determine what type of work and which users or security labels you want RACF to validate.
- "Understanding Mixed Security Environments" on page 458 explains how different levels of JES and RACF affect security processing.
- "Authorizing Jobs" on page 458 explains how you can use RACF to validate inbound jobs from other nodes in a network.
- "Controlling User ID Propagation in a Local Environment" on page 443 explains how you can use RACF to validate inbound jobs from other nodes in a network when the installation does not use the NODES profile.
- "Using Submitter Information During Job Verification" on page 460 explains how you can use RACF to validate inbound jobs using the submitter's security information from the NJE environment.
- "Authorizing SYSOUT" on page 461 explains how you can use RACF to validate inbound job output (SYSOUT) from other nodes in a network.
- "Validating SYSOUT Based on the Submitter" on page 464 explains how RACF can be used to validate inbound work using the submitter's security information instead of the owner's security information.
- "Translating Security Information" on page 465 explains how RACF can be used to replace inbound user IDs, group names, or security labels with locally-defined values.
- "Understanding Default User IDs" on page 467 describes how JES handles security information for work from incompatible nodes and explains how JES handles security information that belongs to store-and-forward work. This section also describes how you can use RACF to manipulate default security information.
- "How JES Sends Security Information" on page 468 explains where JES obtains missing security information for NJE work.
- "Defining Profiles in the NODES Class" on page 469 shows you how to set up the protection defined in the profiles and how to activate not only the NODES class but also the SETROPTS RACLIST processing for the class.
- "Defining Nodes as Local Input Sources" on page 469 explains how you can use RACF to treat SYSOUT from another node as if the SYSOUT originated at the home node.

Understanding NODES Profiles

You can use profiles in the NODES class to control how RACF validates inbound work on an NJE network. As with other RACF profiles, a NODES profile consists of a profile name, a profile class, a universal access authority, and an ADDMEM value. The profile name is a three-part identifier that indicates the origin of the work and the type of security information you want to validate. The universal access authority

determines the actions that RACF performs on the inbound work. This information is described in Table 34 on page 459 and Table 35 on page 463.

Note: Access lists do not apply to NODES class profiles. The ADDMEM value is used to translate to locally-defined values.

A NODES profile name has the following format:

nodename.worktype.name

where:

nodename Is the name of the node from which you expect inbound work. For jobs, this is the submitting node. For SYSOUT, this is the execution node.

Notes:

1. If &SUSER is specified as an ADDMEM value in a profile that controls SYSOUT, a second check is done where *nodename* is the submitting node.
2. If &DFLTGRP is specified as an ADDMEM value in a profile that deals with groups (either jobs or SYSOUT), the user's default group is used.
3. It is recommended that you define a profile in the RACFVARS class named &RACLNDE, and use &RACLNDE for all nodes that are considered local to your system. For more information, see "Setting Up NODES Profiles" on page 455.

worktype Is the type of work to be controlled by the profile.

Notice that the *last character,* J or S, indicates the type of work to be validated. J indicates jobs; S indicates SYSOUT.

RUSER Controls commands originating from NJE nodes. The *nodename* is used as the name on the third qualifier.

USERJ Controls jobs by the user ID specified on the third qualifier. The job is controlled by who the submitter is. This type of profile is also used to determine the amount of trust the job has. For details, see "Understanding Mixed Security Environments" on page 458.

USERS Controls SYSOUT by the user ID specified on the third qualifier. The SYSOUT is controlled by who the owner is. This type of profile is also used to determine the amount of trust the SYSOUT has. For details, see "Understanding Mixed Security Environments" on page 458.

GROUPJ Controls jobs by the group name specified on the third qualifier.

GROUPS Controls SYSOUT by the group name specified on the third qualifier.

SECLJ Controls jobs by the security label specified on the third qualifier.

SECLS Controls SYSOUT by the security label specified on the third qualifier.

For example, a value of USERJ specifies that you want RACF to use the profile to validate inbound jobs; a value of USERS specifies that you want RACF to use the profile to validate inbound SYSOUT.

name Is the actual user ID, group name, or security label you want validated. If you are using NODES profiles to allow the use of these input values, you must either define these values in your RACF database or use the ADDMEM operand to translate them into acceptable values for your system. For jobs, the submitter information is substituted. For SYSOUT, the owner information is used. (See "Understanding Mixed Security Environments" on page 458.)

For example, the following profile controls whether jobs coming from user ID WAYNE at node BERMUDA can be executed here:

```
BERMUDA.USERJ.WAYNE
```

You can optionally associate a local user ID with user ID WAYNE by specifying the user ID on the ADDMEM operand.

You can specify generic characters in the profile name to control a wider range of work. For example, if you place an asterisk in place of the *nodename* value, RACF performs the requested type of validation for work from all nodes in the network (unless a more specific profile exists). Examples of generic profiles in the NODES class are shown later in this chapter. For more information, see "Choosing Between Discrete and Generic Profiles in General Resource Classes" on page 199.

If you installed RACF and did not activate the NODES class, JES validates jobs and SYSOUT in the following manner:

- JES runs only those jobs that are destined for your node and that have a valid user ID and password on the job card if BATCHALLRACF is active. If BATCHALLRACF is not active, the job can run without a RACF user ID.
- A security label of SYSHIGH is assigned to all SYSOUT destined for your node (if security labels are being used) and may be printed only on those devices permitted to SYSHIGH data. JES assigns the default user ID to this SYSOUT. For information about default user IDs, see "Understanding Default User IDs" on page 467.
- All work destined for another node remains unchanged.

If you choose to activate the NODES class, you must gather information from your JES system programmer so that you can set up profiles to control the work entering your system. The following sections identify the appropriate values for each type of work.

Setting Up NODES Profiles

To set up NODES profiles, you must activate the RACFVARS class first, issue SETROPTS RACLIST, and, if you are going to define generics, make sure that SETROPTS GENERIC is active for the RACFVARS class. You should consider the following approach to setting up NODES profiles:

1. Define a profile for each node for which you wish to control inbound work. (If you have several nodes that you are treating identically, consider creating RACFVARS profiles and using the RACF variables in NODES profile names. This can reduce the number of NODES profiles that you must maintain.)
2. Define a "top" generic profile to control all work not controlled by more specific NODES profiles.

3. For each node, define profiles with USER*x*, SECL*x* or GROUP*x* qualifiers only if you want to:
   - Prevent work with the specified user ID, security label, or group name from entering your node (determined by the UACC of the profile).
   - Translate the specified user ID, security label, or group name to a local value (specify the ADDMEM operand to do this).
4. Define the local node or nodes in the &RACLNDE profile in the RACFVARS class. Enter:

   ```
   RDEFINE RACFVARS &RACLNDE ADDMEM(nodea nodeb...)
   ```

   In effect, this allows security information to be accepted for verification without the use of NODES profiles. That is, the information is used as passed because it is considered local.

   For SYSOUT, this allows the owner information to be used without a NODES lookup, or automatically allows the submitter to become the SYSOUT owner when &SUSER is used (see “How SYSOUT Requests Are Verified” on page 445).

   For jobs, this allows the special JES2 pre-execution reroute case to use the information as passed without translation, and allows the spool unload and reload of jobs to propagate the information automatically without requiring NODES profiles. See “Defining Nodes as Local Input Sources” on page 469.

   Note: Group names are not propagated when the node is defined to &RACLNDE The execution-user’s default group is used.
5. If an inbound job has been submitted as a surrogate job on its originating system (see “Allowing Surrogate Job Submission” on page 441), the PASSWORD parameter is not specified on its JOB statement. Therefore, you must specify UACC(CONTROL) or higher in the NODES profile controlling such jobs, or UACC(UPDATE) or higher if the job is from an uplevel node to prevent requiring password verification (see “Understanding Mixed Security Environments” on page 458).

RACSLUNK SECLABEL: RACSLUNK is a special SECLABEL, assigned to SYSOUT, that is received from another node if the SECLABEL class is active on the receiving node and the NJE header contains an unknown or blank SECLABEL.

Because the RACSLUNK SECLABEL is not predefined, no one is able to access the SYSOUT. This situation can be handled in one of two ways:

- Define the RACSLUNK to RACF as a valid SECLABEL and authorize users to access it.
- Use the NODES class translation facility to translate it to a known SECLABEL.

For an exercise to learn which NODES profiles are used, see Figure 51 on page 457.

Assume the following profiles:

```
(1)   POKMVS.SECLJ.A      ADDMEM(ALPHA)    UACC(READ)
(2)   POKMVS.SECLS.A      ADDMEM(ALPHA)    UACC(READ)
(3)   POKMVS.SECL%.A                       UACC(NONE)  /*never used*/
(4)   POKMVS.USERJ.JOHN   ADDMEM(JOHNNY)   UACC(UPDATE)
(5)   POKMVS.USERS.JOHN   ADDMEM(JOHNNY)   UACC(UPDATE)
(6)   POKMVS.USER%.JOHN                    UACC(NONE)  /*never used*/
(7)   POKMVS.USER%.TOM                     UACC(NONE)
(8)   POKMVS.USER%.*      ADDMEM(NONAME)   UACC(UPDATE)
(9)   POKMVS.*.*          ADDMEM(X)        UACC(READ)
(10a) *                                    UACC(NONE)
(10b) *.USERJ.*                            UACC(NONE)
```

1. If a job is submitted from user JOHN at node POKMVS with SECLABEL A, profiles (1), (4), and (9) are used.
   Profile (4) translates the user ID to JOHNNY.
   Profile (9) translates the group name to X (there is no profile with the GROUP operand.)
   Profile (1) translates the SECLABEL to ALPHA.
2. Profile (3) would never be used because profiles (1) and (2) are discrete profiles that cover all work from node POKMVS that has security label A.

   Profile (6) would never be used because profiles (4) and (5) are discrete profiles that cover all work from user JOHN at node POKMVS.
3. If jobs or SYSOUT come in from user TOM at POKMVS, profile (7) fails the job or purges the output.
4. If a job comes in from anyone other than JOHN or TOM at POKMVS, with SECLABEL A, profiles (1), (8), and (9) are used.
   Profile (8) translates the user ID to NONAME.
   Profile (9) translates the group name to X (there is no profile with the GROUP operand.)
   Profile (1) translates the SECLABEL to ALPHA.

   Note: Profile (8) translates many user IDs to one. You might do this to create a guest user ID that can be used by any otherwise unknown user coming in from POKMVS. With such a user ID, you can allow people from POKMVS to access certain resources without having to give each of them a user ID on your system.
5. Because there is no POKMVS profile with the GROUP operand, profile (9) is the generic that is used to translate group names. Therefore all jobs and SYSOUT that come from POKMVS get group X. (If profile (9) did not have ADDMEM specified, there would be no translation of group names.)

   Also, all security labels from POKMVS, except security label A, are translated to X.
6. Profile (10a) fails all NJE jobs and SYSOUT for any other user, group, or security label that is not covered by a more specific NODES profile. If you want to have just default control for any NJE jobs, and not control SYSOUT, use profile (10b) instead.

Figure 51. Which NODES Profiles Are Used?

Understanding Mixed Security Environments

Your network may be a mixed environment, that is, it can contain nodes in which different levels of JES and RACF, or non-JES systems, are installed. Networking in a mixed environment causes JES and RACF to validate work differently in some cases. For example, certain security information, such as security labels, may not be sent with work from some systems. The following list categorizes the various environments into three groups:

- Uplevel security systems
  - Systems running z/OS or OS/390 where RACF is installed and active.
- Downlevel security systems
  - Systems running z/OS or OS/390 where RACF is not used but another security product is being used.
  - Systems not running z/OS or OS/390.
- Default security systems
  - Systems running z/OS or OS/390 where no security product is being used.

The terms (uplevel, downlevel, and default) describe the security systems of the source nodes. This tells JES and RACF how much of the security information has been verified at the source. They are used to determine how much the receiving node trusts the source nodes.

For jobs, the amount of trust determines under what circumstances RACF propagates the submitter. For SYSOUT, it determines under what circumstances RACF accepts the owner information. NJE uses the NODES profile UACC to determine level of trust. Use these definitions when the following sections in this chapter refer to uplevel, downlevel, and default security systems. See Table 34 on page 459 and Table 35 on page 463.

Authorizing Jobs

You can control which network jobs are authorized for processing at your installation on the basis of submitter's user ID, group name, or security label associated with the inbound job.

To authorize or restrict jobs entering your system from another node, define a NODES profile that specifies the criteria on which jobs are accepted. Ask your JES system programmer for the following:

- The node names from which you expect jobs
- The user IDs or group names from which you expect jobs
- The security labels that you should accept
- The universal access authority, which determines how JES3 processes the job. Table 34 on page 459 lists the universal access authorities you can assign and defines the validation that RACF performs.

Table 34. NODES Class Operands and the UACC Meaning for Inbound Jobs

Type of Check (operand)	UACC			
	NONE	**READ**	**UPDATE**	**CONTROL or greater**
User ID (USERJ)	Fails the job.	Verifies all security information available including password validation.	If the job is from an uplevel node, that is, if a non-default valid security token is passed, propagates the submitter's or translated security information as the owning security information without password validation. See "Understanding Mixed Security Environments" on page 458. If the job is from a default or downlevel node, processing is the same as for a UACC of READ.	Same as UPDATE, but default security or downlevel information is allowed. CONTROL allows a downlevel system to send jobs to your node without passwords. RACF does not validate passwords. See "Understanding Mixed Security Environments" on page 458.
Group name (GROUPJ)	Fails the job.	Translates group name to that specified in ADDMEM. If ADDMEM is not specified, uses the group name received.		
Security Label (SECLJ)	Fails the job.	Translates the submitter's SECLABEL to that specified in ADDMEM. If ADDMEM is not specified, uses the security label received.		
Note: For more details on how NJE jobs are processed, see "Authorizing Jobs" on page 458.				

Note: If no profile exists for a job when the NODES class is active or if the NODES class is inactive, RACF performs only user ID, group name, and password validation without performing any translation.

If no profile exists for a job when the NODES class is active, RACF verifies all security information available and a valid password and user ID must be specified on the job card.

You can further reduce the risk of security exposures by allowing jobs to be submitted from other nodes without requiring a password if the sending node properly validates and transmits a user's identity. You can either allow the submitter's identity (that is, the user ID and security label) to be propagated to the job or you can specify that the submitter is a surrogate submitter who can submit jobs on behalf of other users without needing a password.

For either case, you indicate in NODES class profiles which nodes are trusted to provide valid submitter identity information. You can restrict the trusted information to specified user IDs, group names, or security labels, if desired.

This submitter identity information in combination with user data on the job card is used to determine the user identity to be used for the job.

- If no user ID or password is specified on the job card, the submitter's identity is propagated to the job.[8]
- If a user ID but no password is specified, the user ID is allowed if the submitter is authorized as a surrogate for that user ID.[8]
- If both user ID and password are specified on the job card, the submitter's identity is not propagated to the job, but will still be used for JESJOBS checking. Normal password validation is performed.

Controlling User ID Propagation in an NJE Environment

If an installation does not use NODES profiles for its networking protection (in other words, it uses the RACFVARS profile &RACLNDE to treat all its remote nodes as local), propagation control is the same as for actual local jobs (see "Controlling User ID Propagation in a Local Environment" on page 443). Otherwise, an installation wanting propagation control across the network needs to define one or more NODES profiles, possibly with a RACFVARS profile, similar to the following:

```
RDEFINE RACFVARS &PROPCON ADDMEM(USER25 USER42 USER19 USER22 USER11)
RDEFINE NODES *.USERJ.&PROPCON UACC(READ)
```

Be sure that both the RACFVARS and the NODES classes are active and generics are active for the NODES class, that you bring the classes in storage using SETROPTS RACLIST, and that you issue a SETROPTS RACLIST REFRESH after you define the two profiles. You need these profiles on every receiving system where you want propagation to be controlled. Every user ID that has a PROPCNTL entry on that system should be included in the ADDMEM list for &PROPCON.

With this setup, if a user ID is *not* coded on the job card, the job is routed to another node, and the submitting user ID is a member of &PROPCON on the receiving side, the job runs with the undefined user ID (default of ++++++++), assuming SETROPTS(JES(BATCHALLRACF)) and SETROPTS(JES(XBMALLRACF)) are not in effect.

Note that a better-fitting NODES profile with a higher UACC negates this protection. For example, if in addition to the two profiles above you have a NODES profile NODEX.USERJ.CICS1 with UACC(CONTROL), even if CICS1 is a member of &PROPCON, an incoming job submitted by CICS1 runs as CICS1.

For more information about why you would want user propagation controlled, see "Controlling User ID Propagation in a Local Environment" on page 443.

Using Submitter Information During Job Verification

With NJE jobs, as with local jobs, RACF makes several checks based on the submitter of a job. The submitter information is used during SURROGATE checking (see "Allowing Surrogate Job Submission" on page 441). It helps to ensure that the security label of the job takes precedence over the security level of the submitter. RACF does this check when security labels are being used, taking into consideration the setting of the SETROPTS MLS option. Submitter information is also used for JESJOBS checking during submission of a job (see "Controlling Who Can Submit Jobs by Job Name" on page 448).

With local jobs, the submitter information is used as it is passed to RACF. Normally, it is assumed to be valid. During any of the submitter checks, however, it is subject to reverification. Any incorrect information causes the specific check to fail.

8. In either case, if SECLABEL is specified on the job card, it is used. If not, the SECLABEL of the submitter is propagated to the job.

With NJE jobs, the submitter information used depends on whether the submitting node is trusted. If the submitting node is trusted, the submitter information is either used as passed or translated through NODES profiles. This information is subject to reverification during any submit check that may be performed. This is consistent with local jobs.

If the submitting node is not trusted, the submitter information cannot be used as passed to RACF. When the submitter is identified by token information, the submitter is then represented by the NJE unknown user (that is, no user ID). The original submitter information is discarded. This allows UACC access to the checks made on behalf of the submitter, such as SURROGATE and JESJOBS.

RACF validates an NJE batch job based on the submitter node and submitter user ID in a USERJ profile and on the submitter node and submitter group name in a GROUPJ profile. If there is an ADDMEM value, the NJE batch job submitter user ID is translated to the ADDMEM value before the validation checks are made.

When RACF determines that a job is not from a trusted node, the submitter user ID of the NJE batch job is set to the NJE unknown user ID and the submitter group name is changed to blanks. For a job that is submitted from a trusted node, the translated submitter user ID is propagated and becomes the user ID with which the NJE batch job runs.

USERJ NODES profiles are checked before the GROUPJ NODES profiles. After successful verification based on the submitter node and user ID, GROUPJ NODES profiles are used to validate NJE batch jobs, based on the submitter node and group name. If there is an ADDMEM value, the NJE batch job submitter group name is translated to the ADDMEM value before the validation checks are made.

Note: If no USERJ NODES profile exists, the GROUPJ NODES profile is not checked.

A GROUPJ NODES profile can be used to fail incoming jobs based on the submitter's group by specifying UACC of NONE in the profile. A GROUPJ NODES profile can also be used to translate the submitting group to an appropriate group for the receiving system. This is done by specifying a UACC of at least READ and an appropriate ADDMEM member.

If the installation does not want incoming jobs to fail based on the groups, a special ADDMEM of &DFLTGRP can be used. This is not a RACFVARS variable. It just specifies that for jobs matching this GROUPJ profile, the resulting user's default group should be used in the verification.

Example:

```
RDEFINE NODES Z.GROUPJ.* UACC(READ) ADDMEM(&DFLTGRP)
```

Assuming appropriate use of USERJ profiles, all NJE batch jobs from node Z will have SURROGAT and JESJOBS checking done based on the submit-user's default group. Checking done on the execution-user (assuming the submit group is propagated, that is, GROUP is not on the jobcard), will be done with the execution-user's default group.

Authorizing SYSOUT

You can control the processing of SYSOUT at your installation based on the user ID, group name, or security label associated with the inbound SYSOUT. If no profile

exists for an NJE SYSOUT when the NODES class is active, SYSOUT ownership cannot be assigned. See “Understanding Default User IDs” on page 467.

To authorize or restrict SYSOUT entering your system from another node, define NODES class profiles that identify the criteria on which SYSOUT is accepted. Ask your JES system programmer for the following:

- The node names from which you expect SYSOUT
- The user IDs, group names, and security labels from which you expect SYSOUT
- The universal access authority, which determines how JES processes the SYSOUT. RACF can assign ownership based on either the user ID and node that created the SYSOUT or the user ID and node that submitted the job that created the SYSOUT.

  Notes:

  1. If the NODES profile allows the user ID to be associated with the SYSOUT, but the user security information is incorrect, an IRR808I message is issued and processing continues with the NJE unknown user as set by SETROPTS JES(NJEUSERID(*userid*)).
  2. &DFLTGRP can be used for SYSOUT in the same way as in batch jobs. Specifying ADDMEM of &DFLTGRP in a GROUPS profile will cause verification to be done on the owning-user’s default group.

  Table 35 on page 463 lists the universal access authorities you can assign and defines the validation that RACF performs.

Table 35. NODES Class Operands, UACC and SYSOUT Ownership When Node Is Not Defined to &RACLNDE

Type of Check (operand)	UACC			
	NONE	**READ**	**UPDATE**	**CONTROL or greater**
User ID (USERS)	**Check of User ID and Node That Created SYSOUT**			
	Purges the output.	If the translation value from ADDMEM is &SUSER, check submitting user ID and node. Otherwise, assigns ownership of the output to the default NJE user ID (default is ????????).	If default or no security information is available, that is, from a downlevel or default node, processing is the same as a UACC of READ. If security information is from an uplevel node, that is, a non-default valid security token is passed, assigns the translation value from ADDMEM to the output. When ADDMEM is not specified, ownership is assigned to the user ID that created the output. See “Understanding Mixed Security Environments” on page 458. If the translation value from ADDMEM is &SUSER, check submitting user ID and node.	Processing is similar to UACC(UPDATE) except RACF translates any available information from any type of security system. This allows RACF to assign local user IDs to output from downlevel systems. See “Understanding Mixed Security Environments” on page 458. If the translation value from ADDMEM is &SUSER, check submitting user ID and node.
	Check of Submitting User ID and Node (Only When &SUSER Is Specified for ADDMEM)			
	Assigns ownership of the output to the default NJE user ID (default is ????????).	Assigns ownership of the output to the default NJE user ID (default is ????????).	Assigns ownership of the output to the default NJE user ID (default is ????????).	Assigns the translation value from ADDMEM to the output, if available. If the translation value from ADDMEM is &SUSER, assigns the submitting user ID to the output. Otherwise, assigns ownership of the output to the default NJE user ID (default is ????????).
	Note: When you specify &SUSER for ADDMEM and the submitting node *is* defined to &RACLNDE, ownership is assigned to the submitter. See “How SYSOUT Requests Are Verified” on page 445.			
Group Name (GROUPS)	Purges the output	Translates group name to that specified in ADDMEM. If ADDMEM is not specified, uses the group name received.		
Security Label (SECLS)	Purges the output	Translates SECLABEL to that specified in ADDMEM. If ADDMEM is not specified, uses the security label received.		

Table 35. NODES Class Operands, UACC and SYSOUT Ownership When Node Is Not Defined to &RACLNDE (continued)

Type of Check (operand)	UACC			
	NONE	READ	UPDATE	CONTROL or greater

Notes:

1. If the node name is specified in the RACFVARS profile named &RACLNDE, the node is treated as a locally attached node and RACF verifies the supplied security information.
2. For more details on how NJE SYSOUT is processed, see "Authorizing SYSOUT" on page 461 and "Validating SYSOUT Based on the Submitter".

Validating SYSOUT Based on the Submitter

JES normally validates SYSOUT based on the owner's security information. The owner's security information accompanies each piece of SYSOUT as it travels through the network.

You can define profiles that cause RACF to assign ownership of the SYSOUT to the submitter. For example, you can allow a user to submit a job to another node, have the job execute under another user ID, and allow the submitting user to view the output on its return.

To translate inbound SYSOUT ownership to the submitter, specify &SUSER as the value on the ADDMEM operand of the NODES profile.

This works with potentially multiple NODES profiles as follows:

First, the NODES profile is used that matches the form *execution-node.USERS.userid*. If the UACC is not NONE and the ADDMEM is &SUSER, a check is made to see if the submitter is set up to be the owner. If the submit node is found to be a member of the RACFVARS &RACLNDE profile, the submitter user ID and group are associated with the SYSOUT without change. This is because the submit node is considered local.

If the submit node is not local in this way, a second NODES profile that matches the form *submit-node.USERS.submitter-id* is used; and, if the UACC is CONTROL and there is an ADDMEM value, the submitter values are associated with the SYSOUT. If the ADDMEM value is not &SUSER, the ADDMEM value is used as the SYSOUT owner user ID.

If the ADDMEM is &SUSER, the original submitter is used as the SYSOUT owner user ID. The second NODES profile cannot be used to purge SYSOUT. The first NODES profile has already established the level of trust and the second NODES profile is used only for determining the owning user ID of the SYSOUT. A UACC of NONE on the second NODES profile assigns the ??????? userid. For more details, see Table 35 on page 463.

When associating the submitter with the SYSOUT in the non-local case, a third NODES profile can be used that matches the form *submit-node.GROUPS.submit-group*. If this profile exists and has an ADDMEM value, the ADDMEM value is used as the SYSOUT owner group, regardless of the UACC. Otherwise, the original submit group is associated with the SYSOUT. Verification of the SYSOUT continues with the owner values altered as described above.

Translating Security Information

You can avoid having to maintain identical user IDs, group names, and security labels in RACF databases throughout a network by translating inbound user IDs, group names, and security labels into predefined values defined at your node.

Use the ADDMEM operand on the RDEFINE or RALTER command to specify the translation values for inbound security information. For example, if you want all inbound work with a security label of VERYCONF to be translated to a security label of NOLOOKAT at your system, enter:

```
RDEFINE NODES *.SECL*.VERYCONF ADDMEM(NOLOOKAT) UACC(READ)
```

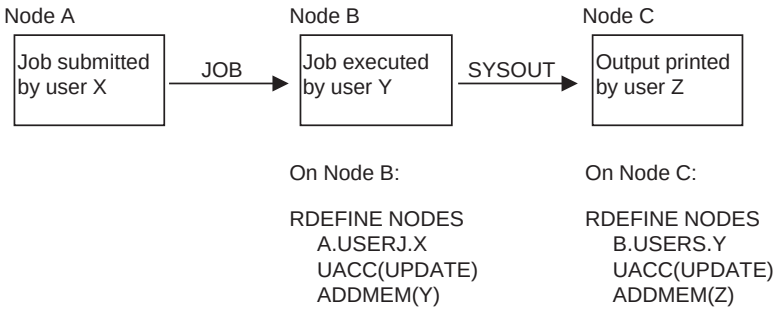
Notes:

1. You should specify only one value on the ADDMEM operand. Any subsequent values are ignored.
2. For jobs, an ADDMEM of &SUSER is ignored, as the NODES profile lookup for jobs automatically deals with submitter information. It would be treated as though no ADDMEM were specified for the profile. For more information on &SUSER, see “Validating SYSOUT Based on the Submitter” on page 464.

If you do not define profiles that translate inbound user IDs, group names, and security labels, those inbound values must be defined in your RACF database or the work does not pass RACF validation.

Note: If the SECLABEL class is not active on your system, inbound security labels are ignored.

Example: Simple NJE User Translation: Figure 52 shows how user IDs are translated.



User X is known as user Y on node B, and user Z on node C. In this example, user X on node A submits a job that runs on node B. The output is printed on node C under user ID Z.

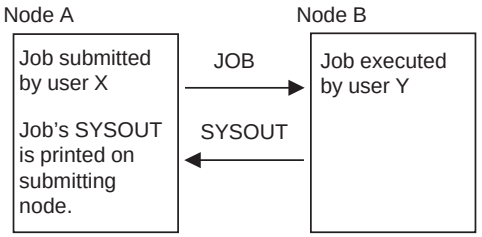
On node B, the existence of a profile named A.USERJ.X, with ADDMEM(Y) specified, causes RACF to translate the user ID of jobs from user X at node A. On node B, such jobs run as if submitted by user Y.

On node C, the existence of a profile named B.USERS.Y, with ADDMEM(Z) specified, causes RACF to translate the user ID of SYSOUT from user Y at node B. On node C, such SYSOUT can print as if submitted by user Z.

Figure 52. Example: Simple NJE User Translation

Example: Simple NJE User Translation Using &SUSER: Figure 53 shows how user IDs are translated when &SUSER is specified on the ADDMEM operand. This can be useful when jobs are run on a remote system, but the output is printed on the submitter's system.

Note: If you wish, you could specify &SUSER on a third system (as in node C in Figure 52 on page 465).



On Node A:

```
RDEFINE NODES
   B.USERS.Y
   UACC(UPDATE)
   ADDMEM(&SUSER)

RDEFINE NODES
   A.USERS.X
   UACC(CONTROL)
   ADDMEM(&SUSER)
```

On Node B:

```
RDEFINE NODES
   A.USERJ.X
   UACC(UPDATE)
   ADDMEM(Y)
```

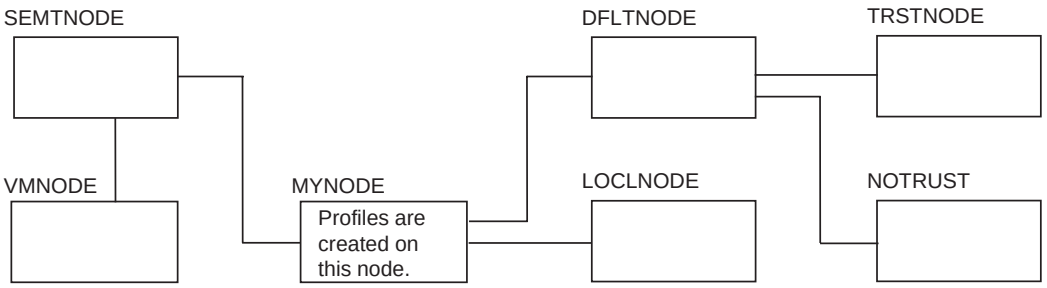
User X is known as user Y on node B. In this example, user X on node A submits a job that runs on node B. The output is returned to user X's home node (node A) to be printed.

On node B, the existence of a NODES profile named A.USERJ.X with UACC(UPDATE) and ADDMEM(Y) means that jobs from user X at node A are to be executed under user Y.

On node A, the existence of a NODES profile named B.USERS.Y with UACC(UPDATE) and ADDMEM(&SUSER) means that SYSOUT from user Y at node B is to be owned by the user who originally submitted the job, provided the second lookup is successful. The second lookup involves the submitter of the job (X) and the node that he submitted it from (A). Therefore, on node A, the existence of a NODES profile named A.USERS.X with UACC(CONTROL) and ADDMEM(&SUSER) means that the SYSOUT is to be owned by user X.

Figure 53. Example: Simple NJE User Translation Using &SUSER

Example: Trusted, Semitrusted, and Untrusted Nodes: Figure 54 on page 467 shows a sample NJE network in which some nodes are trusted (see "Understanding Mixed Security Environments" on page 458), some nodes are semitrusted (verification is done on inbound work), and some nodes are not trusted (no inbound work is allowed to run).



In this example, profiles on node MYNODE control inbound work as follows:

Trusted Nodes

```
RDEFINE NODES TRSTNODE.USER%.* UACC(UPDATE)
RDEFINE NODES LOCLNODE.USER%.* UACC(UPDATE)
RDEFINE NODES VMNODE.USER%.* UACC(CONTROL)
```

Semitrusted Nodes

```
RDEFINE NODES SEMTNODE.USER%.* UACC(READ)
RDEFINE NODES DFLTNODE.USER%.* UACC(READ)
```

Untrusted Node

```
RDEFINE NODES NOTRUST.*.* UACC(NONE)
```

Note: To prevent any unknown nodes from submitting work to be done on your node, create the following profile:

```
RDEFINE NODES *.*.* UACC(NONE)
```

Figure 54. Example: Trusted, Semitrusted, and Untrusted Nodes

Understanding Default User IDs

RACF assigns a default user ID to all work that enters your node when:

- SYSOUT enters from one of the following:
  - A downlevel node
  - A default node

  For more information, see “Understanding Mixed Security Environments” on page 458.
- SYSOUT or a job enters your node, but your node is an intermediate (store-and-forward) node on the path to the work’s final destination. The default user ID protects work while it resides on spool awaiting transmission.
- SYSOUT enters your node when the NODES class is active and no applicable USERS profile exists.

RACF uses eight question marks (????????) as the user ID for all inbound work meeting the above criteria. RACF also assigns the default user ID to all store-and-forward work that resides temporarily at your node. The default user ID protects work while it resides on spool.

You cannot directly permit the default user ID (???????? or installation-defined) to any resources. However, you can translate the default user ID to a valid user ID if you want to process any of this type of work at your system.

You can change the ???????? user ID by using the NJEUSERID operand on the SETROPTS command:

```
SETROPTS JES(NJEUSERID(?NETWORK))
```

The user ID you specify on the NJEUSERID operand cannot be a user ID defined in the RACF database. Also, if you specify a user ID on the NJEUSERID operand, you cannot later define a user profile for that user ID. This prevents network jobs from having access to RACF-protected resources on your system.

The following example shows how to do this for jobs:

```
RDEFINE NODES nodename.USERJ.???????? UACC(READ or higher) ADDMEM(NJEJOBS)
```

The following example shows how to do this for SYSOUT:

```
RDEFINE NODES nodename.USERS.???????? UACC(UPDATE or higher) ADDMEM(NJESOUT)
```

The following example shows how to do this for both SYSOUT and jobs:

```
RDEFINE NODES nodename.USER%.???????? UACC(READ or higher) ADDMEM(NJEWORK)
```

Note: This example assumes that a SETROPTS GENERIC(NODES) was previously issued to turn generics on for this class and that a SETROPTS REFRESH was then done.

You would also need to create user profiles for the translated user IDs (NJEJOBS, NJESOUT, or NJEWORK), and permit the user IDs to appropriate resource profiles (or connect them to appropriate groups).

Local jobs that enter the system without a user ID are assigned a user ID of ++++++++ (8 plus signs). You can specify which user ID to assign to such jobs by entering the following command:

```
SETROPTS JES(UNDEFINEDUSER(userid))
```

Note: The user ID you specify on the UNDEFINEDUSER operand cannot be a user ID defined in the RACF database. Also, if you specify a user ID on the UNDEFINEDUSER operand, you cannot later define a user profile for that user ID. This prevents undefined users from having access to RACF-protected resources on your system.

However, these user IDs can be used in JESSPOOL profile names. JES uses these names to associate an owner with the spool data, and to keep logical undefined users from accessing the data of network undefined users.

How JES Sends Security Information

Security information is sent from node to node in an NJE network. When a node receives a job through a network, RACF determines who submitted the job. After determining the submitting user ID, RACF can translate the submitting user ID to a valid user ID on this system if a profile on the receiving node specifies that the user ID must be translated. RACF uses the submitting user ID or its translation to supply any missing security information. Security information that RACF propagates is:

- User ID
- Password
- Security label

Defining Profiles in the NODES Class

To create profiles in the NODES class, take the following steps:

1. Ask your JES system programmer for the information needed to create the profiles. This includes the following:
   - Information for specifying profile names
   - For each profile to be created, the UACC to be specified
   - For each profile to be created, the values to be translated (to be specified on the ADDMEM operand)

   Note: You should work with your JES system programmer to determine which user or group should be specified in the OWNER field of the profiles. This user or group is responsible for maintaining the profiles.
2. Use the RDEFINE command to create the profiles. For examples, see Figure 51 on page 457, Figure 52 on page 465, Figure 53 on page 466, and Figure 54 on page 467.
3. When you are ready to start using the protection defined in the profiles, activate the NODES class and activate SETROPTS RACLIST processing for the class. You can do these two actions in one command:

   ```
   SETROPTS CLASSACT(NODES) RACLIST(NODES)
   ```

   Notes:

   a. Any time you make a change to a NODES profile, you must also refresh SETROPTS RACLIST processing for the NODES class for the change to take effect.

   b. RACF does not do any logging nor issue any messages for the NODES class.

Defining Nodes as Local Input Sources

You can use RACF to treat nodes the same as locally attached devices. To do this, use the &RACLNDE profile in the RACFVARS class to identify the nodes that you want RACF to consider as local. See “Setting Up NODES Profiles” on page 455. A node name defined to &RACLNDE either shares your RACF database or is the value you are using to rename your node when you are changing node names.

To do this take the following steps:

1. Ask your JES system programmer for the names of the nodes to be treated as local.
2. Define profile &RACLNDE in class RACFVARS:

   ```
   RDEFINE RACFVARS &RACLNDE UACC(NONE)
   ```
3. Using the ADDMEM operand on the RALTER command, identify which nodes are to be treated as local nodes:

   ```
   RALTER RACFVARS &RACLNDE ADDMEM(node1 node2 node3 ...)
   ```

   Notes:

   a. If you define a node as a local node, you must ensure that its RACF database is identical to the one on your node.

   b. Because there are no defaults for &RACLNDE profiles in the RACFVARS class, you must identify your own local node using the ADDMEM operand.
4. When you are ready to start using the protection defined in the profiles, activate the RACFVARS class and activate SETROPTS RACLIST processing for the class. You can do these two actions in one command:

   ```
   SETROPTS CLASSACT(RACFVARS) RACLIST(RACFVARS)
   ```

Notes:

a. Any time you make a change to a RACFVARS profile, you must also refresh SETROPTS RACLIST processing for the RACFVARS class for the change to take effect.
b. This also activates other functions that are administered through the RACFVARS class.

Authorizing Outbound Work

You can use the WRITER class to control whether work is authorized for transmission to a specific NJE node. To do this, create profiles in the WRITER class that have profile names with the following format:

jesname.NJE.*nodename*

For more information, see “Controlling Where Output Can Be Processed” on page 480.

Using Security Labels to Control Writers

If both the WRITER and the SECLABEL class are active when RACF checks a writer’s authority to process outbound work, RACF uses “reverse MAC” (mandatory access checking). That is, the outbound work must have a security label, and the security label of the writer must be equal to or greater than the outbound work’s security label.

You can use this to limit the sensitivity of the data that writers can process. For example, if you have some writers that process low-sensitivity information, you can assign those writers a low-sensitivity security label, such as SYSLOW. This prevents sensitive work from leaving your node through those writers.

Controlling Access to Spool Data

You can use RACF to provide security for your spool data, including:

- Access to spool data
- Data sets dumped from spool
- Data sets restored to spool
- JESNEWS
- SYSIN data sets
- SYSOUT data sets
- SYSLOG
- Trace data sets (for JES2).

The following sections identify which spool resources you can protect, why you may want to protect each resource, and what information you must gather from your JES system programmer so that you can implement RACF protection.

Protecting Data Sets on Spools

You can use RACF to provide access to data sets that reside on spool, including spool files that JES appends to job output, such as JESNEWS. Using RACF allows users other than the owner of a data set to read, copy, print, or delete sensitive job data.

To enable RACF protection of spool data sets, activate the JESSPOOL class:

```
SETROPTS CLASSACT(JESSPOOL)
SETROPTS GENERIC(JESSPOOL)
```

Profiles are not required in the JESSPOOL class for protection to be in effect because the default for the class is failure when no profiles exist. IBM recommends that you activate the generics for the JESSPOOL class because the profile names are system generated.

Notes:

1. When the JESSPOOL class is active, RACF ensures that only authorized users obtain access to job data sets on spool. Authorization to job data sets is provided through RACF user profiles. If there is no profile for a data set, only the user that created the data set can access, modify, or delete it.
2. While a job is executing, RACF optionally audits actions against SYSIN and SYSOUT data sets. For SYSIN data sets, JES invokes RACF each time a SYSIN data set is allocated, opened, or deleted. For SYSOUT data sets, JES invokes RACF each time a SYSOUT data set is created, opened, deleted, or selected for output.
3. For output selection, a data set can be selected by a TSO user through the TSO OUTPUT command. A profile must exist to enable users other than the creator to access data sets using the TSO OUTPUT command.
4. External writers, which are usually started tasks that process output to special devices (such as microfiche), require at least ALTER access to the spool data sets they process. If your installation has external writers, and you activate the JESSPOOL class, you must either ensure that the external writers have ALTER access to appropriate JESSPOOL profiles, or define the external writers as a started procedure with the trusted attribute. You can define them either in the STARTED class or in the RACF started procedures table (ICHRIN03). Otherwise, the external writers cannot process output. Because external writers are installation-written programs, you are strongly recommended to avoid giving them the trusted attribute.
5. If SDSF is installed on your system, JESSPOOL profiles control which action characters and overtypeable fields users can enter on SDSF panels. For complete information on creating JESSPOOL profiles for use with SDSF, see *z/OS SDSF Operation and Customization*.
6. SYSOUT application program interface (SAPI) applications, which are usually started tasks that process output to special devices (like microfiche), require at least UPDATE access to the spool data sets they process. If your installation has SAPI applications, and you activate the JESSPOOL class, you must either ensure that the SAPI applications have UPDATE access to appropriate JESSPOOL profiles, or define the applications as a started procedure with the trusted attribute. You can define them either in the STARTED class or in the RACF started procedures table. Otherwise, the SAPI applications cannot process output.

Defining Profiles for SYSIN and SYSOUT Data Sets

Activating the JESSPOOL class provides protection for SYSIN and SYSOUT data sets. However, you might want to allow specific users to see or work with the SYSIN and SYSOUT data sets created by other users. To do this, take the following steps:

1. Create JESSPOOL profiles for the spool data sets:

   ```
   RDEFINE JESSPOOL profile-name UACC(NONE)
   ```

   where *profile-name* is a 6-part name with the following format:

   ```
   local-nodename.userid.jobname.jobid.dsnumber.name
   ```

   where:

local-nodename
is the name of the node on which the SYSIN or SYSOUT data set currently resides. The local node name appears in the JES job log of every job.

Note: It is recommended that you define a profile in the RACFVARS class named &RACLNDE, and use &RACLNDE for all profiles in the JESSPOOL class.

userid is the user ID associated with the job. This is the user ID RACF uses for validation purposes when the job runs.

jobname is the name that appears in the name field of the JOB statement.

jobid is the job ID assigned to the job by JES. The job ID appears in notification messages and the JES job log of every job.

dsnumber is the unique data set number JES assigned to the spool data set. A “D” is the first character of this qualifier.

name is the name of the data set specified in the DSN= parameter of the DD statement. This name cannot be JESYSMSG, JESJCLIN, JESJCL, or JESMSGLG and follows the naming conventions for a temporary data set. For the temporary data set naming conventions, see *z/OS MVS JCL Reference*. If the JCL did not specify DSN= on the DD statement that creates the spool data set, JES uses a single question mark (?).

Note: You can specify generic characters for any of the qualifiers in the profile name. For example, you can substitute an asterisk (*) for one of the qualifiers, such as *jobid*, if it is not known.

A sample JESSPOOL profile name could be as follows. If user MYUSER submits a job named MYJOB to run on NODEA, and JES assigns a job ID of JOB08237, and the value of DSN= for a SYSOUT data set is OUTPUT, the profile name for a SYSOUT data set created by this job could be:

```
NODEA.MYUSER.MYJOB.JOB08237.D0000112.OUTPUT
```

If job MYJOB is run several times, and the same protection is desired for the OUTPUT data set each time, the profile name could be:

```
NODEA.MYUSER.MYJOB.*.*.OUTPUT
```

2. Give users the appropriate access authority, as follows:

```
PERMIT profile-name CLASS(JESSPOOL)
       ID(userid|groupname)
       ACCESS(access-authority)
```

where *access-authority* is one of the following:

NONE Gives the user *no* access.

READ Lets the user view the spool data set, but does *not* let the user change the data set’s contents or attributes. For example, READ does *not* allow the following operands on the TSO OUTPUT command: DELETE, DEST, NEWCLASS, NOHOLD, and NOKEEP.

UPDATE Lets the user read or update the contents of a spool data set. UPDATE does not allow the user to change the data set’s

attributes. UPDATE also allows users to update spool data sets opened by an application in the same address space.

CONTROL Is equivalent to UPDATE.

ALTER Lets the user read or update a spool data set or change the attribute of a spool data set. For example, ALTER allows any operand to be specified on the TSO OUTPUT command, including operands for deleting and printing. Also, when specified for a discrete profile, ALTER lets the user change the profile itself.

Note: If SDSF is installed on your system, JESSPOOL profiles control which action characters and overtypeable fields users can enter on SDSF panels. For complete information on creating JESSPOOL profiles for use with SDSF, see *z/OS SDSF Operation and Customization*.

Letting Users Create Their Own JESSPOOL Profiles

Users can create their own JESSPOOL profiles if they have CLAUTH authority to the JESSPOOL class. If your installation decides to put the SETROPTS GENERICOWNER option into effect, you can restrict each user to creating JESSPOOL profiles only for his or her own spool data.

To do this, take the following steps:

1. Issue this command:

   ```
   SETROPTS GENERICOWNER
   ```

2. To prevent all users except the system administrator from being able to create JESSPOOL profiles, issue either of the following commands:

   ```
   RDEFINE JESSPOOL ** OWNER(sys_admin_id) UACC(NONE)
   RDEFINE JESSPOOL *  OWNER(sys_admin_id) UACC(NONE)
   ```

3. For each user who should be able to create JESSPOOL profiles for his or her own spool data, create a JESSPOOL profile with the user's user ID specified. Make the user the owner of the profile. For example, for users SMITH and BEN:

   ```
   RDEFINE JESSPOOL nodename.SMITH.** OWNER(SMITH) UACC(NONE)
   RDEFINE JESSPOOL nodename.BEN.**   OWNER(BEN)   UACC(NONE)
   ```

   Note: These examples assume that a SETROPTS GENERIC(JESSPOOL) was previously issued to turn generics on for this class and that a SETROPTS REFRESH was then done.

4. Give users CLAUTH authority to the JESSPOOL class:

   ```
   ALTUSER SMITH CLAUTH(JESSPOOL)
   ALTUSER BEN   CLAUTH(JESSPOOL)
   ```

5. Users with CLAUTH authority can define their own JESSPOOL profiles:

   ```
   RDEFINE JESSPOOL profile-name OWNER(SMITH) UACC(NONE)
   RDEFINE JESSPOOL profile-name OWNER(BEN)   UACC(NONE)
   ```

   where *profile-name* is more specific than the JESSPOOL profile name you defined for this user in step 3.

6. After defining their own JESSPOOL profiles, the users with CLAUTH can use the following PERMIT command to grant other users access to the spool data sets protected by that profile:

   ```
   PERMIT profile-name CLASS(JESSPOOL)
          ID(userid|groupname)
          ACCESS(access-authority)
   ```

where *access-authority* is one of the following:

NONE Gives the user *no* access.

READ Lets the user view the spool data set, but does *not* let the user change the data set's contents or attributes. For example, READ does *not* allow the following operands on the TSO OUTPUT command: DELETE, DEST, NEWCLASS, NOHOLD, and NOKEEP.

UPDATE Lets the user read or update the contents of a spool data set. UPDATE does not allow the user to change the data set's attributes. UPDATE also allows users to update spool data sets opened by an application in the same address space.

CONTROL Is equivalent to UPDATE.

ALTER Lets the user read or update a spool data set or change the attribute of a spool data set. For example, ALTER allows any operand to be specified on the TSO OUTPUT command, including operands for deleting and printing. Also, when specified for a discrete profile, ALTER lets the user change the profile itself.

Note: If SDSF is installed on your system, JESSPOOL profiles control which action characters and overtypeable fields users can enter on SDSF panels. For complete information on creating JESSPOOL profiles for use with SDSF, see *z/OS SDSF Operation and Customization*.

Protecting JESNEWS

JESNEWS is a spool file that contains data to be printed following each job's output. Protecting JESNEWS prevents unauthorized users from adding, modifying, or deleting these files, or (if security labels are used) writing data with a higher security label into these files.

The procedure for protecting JESNEWS depends on whether JES2 or JES3 is installed.

Protecting JESNEWS for JES2

To protect JESNEWS for JES2, take the following steps:

1. Ask the JES2 system programmer for the following information:
   - The fully qualified name of each JESNEWS file to be protected
   - The universal access authority to be associated with each JESNEWS file. For JESNEWS, this value should always be READ to allow all JES users to receive JESNEWS.
   - The user IDs or group names of operators and users that are to be authorized to update JESNEWS. Assign each of these users or groups an access authority of UPDATE to the appropriate profile in the OPERCMDS class. Ensure that all users and operators are defined to RACF.
   - The security label to be associated with each JESNEWS file (if security labels are being used). For JESNEWS, this value should always be the lowest security label (SYSLOW) to allow JESNEWS to be printed for all users.
2. Create the following profiles:

```
RDEFINE JESSPOOL nodename.userid.$JESNEWS.STCtaskid.Dnewslvl.JESNEWS
        UACC(READ)
```

where:

nodename is the name of the node that created the JESNEWS data set.

userid is the user ID associated with your JES2 system.

STC*taskid* is the name of the task that created the JESNEWS data set.

D*newslvl* is the level of this copy of JESNEWS.

For example, for JESNEWS on NODEB:

```
RDEFINE JESSPOOL NODEB.*.$JESNEWS.*.*.JESNEWS UACC(READ)
```

Notes:

a. This example assumes that a SETROPTS GENERIC(JESSPOOL) was previously issued to turn generics on for this class and that a SETROPTS REFRESH was then done.

b. To improve system performance, you should consider including an entry for JESNEWS in the global access checking table. For example:

```
NODEB.*.$JESNEWS.*.*.JESNEWS/READ
```

3. To prevent unauthorized updating of JESNEWS, define a profile in the OPERCMDS class. Any users authorized to update JESNEWS must have ALTER access to this resource:

```
RDEFINE OPERCMDS jesname.UPDATE.JESNEWS UACC(NONE)

PERMIT  jesname.UPDATE.JESNEWS CLASS(OPERCMDS) ID(user or group) ACCESS(ALTER)
```

If RACF is not active, JES2 requests authorization to update JESNEWS from the operator.

Note: If RACF and the SECLABEL class are active, RACF assigns the SECLABEL of the last job that updated JESNEWS to the JESNEWS profile. This could cause jobs with lower security labels than the updating job to miss important information and RACF records security violations for jobs accessing JESNEWS that did not previously occur. To make JESNEWS accessible to all users, the job that creates it should have a SECLABEL of SYSLOW and the data set profile should have a UACC of READ. If the SECLABEL is greater than SYSLOW, JESNEWS does not print in the output of any jobs submitted with a lower SECLABEL.

Protecting JESNEWS for JES3

To protect JESNEWS for JES3, take the following steps:

1. Ask the JES3 system programmer for the following information:
   - The fully qualified name for the JESNEWS file to be protected.
   - The universal access authority to be associated with each JESNEWS file. For JESNEWS, this value should always be READ to allow all JES users to receive JESNEWS.
   - The user IDs or group names of operators and users that are to be authorized to update JESNEWS. Assign each of these users an access authority of UPDATE.
   - The security label to be associated with each JESNEWS file (if security labels are being used). For JESNEWS, this value should always be the lowest security label (SYSLOW) to allow JESNEWS to be printed for all users.
2. Create profiles as indicated by the JES3 system programmer. For example:

```
RDEFINE JESSPOOL node.jesname.JOB00000.D0000000.JNEWSLCL UACC(READ)
RDEFINE JESSPOOL node.jesname.JOB00000.D0000000.JNEWSRJP UACC(READ)
RDEFINE JESSPOOL node.jesname.JOB00000.D0000000.JNEWSTSO UACC(READ)
```

Note: To improve system performance, you should consider including entries for JESNEWS in the global access checking table. For example:

```
node.jesname.JOB00000.D0000000.JNEWSLCL/READ
node.jesname.JOB00000.D0000000.JNEWSRJP/READ
node.jesname.JOB00000.D0000000.JNEWSTSO/READ
```

3. For users who must update JESNEWS, give UPDATE authority:

```
PERMIT profile-name CLASS(JESSPOOL) ID(userid or groupname) ACCESS(UPDATE)
```

Protecting Trace Data Sets (JES2 Only)

For JES2, trace data sets contain information that could compromise your installation's security (for example, user IDs and passwords). You can protect these data sets by defining profiles in the JESSPOOL class and permitting only those users that need access to the data sets.

See the following example:

```
RDEFINE JESSPOOL NODE1.JES.*.*.*.JESTRACE UACC(NONE)

PERMIT  NODE1.JES.*.*.*.JESTRACE CLASS(JESSPOOL) ID(SMITH) UACC(ALTER)
```

Note: This example assumes that a SETROPTS GENERIC(JESSPOOL) was previously issued to turn generics on for this class and that a SETROPTS REFRESH was then done.

Protecting SYSLOG

Your security policy may require that you protect SYSLOG because it is the record of your system's daily activities.

To control SYSLOG, define a JESSPOOL profile for the data set, specifying an appropriate universal access, and then grant access to the user IDs or group names that need a different access.

See the following example:

```
RDEFINE JESSPOOL NODEB.+MASTER+.SYSLOG.*.*.? UACC(NONE)

PERMIT  NODEB.+MASTER+.SYSLOG.*.*.? CLASS(JESSPOOL) ID(SMITH) UACC(ALTER)
```

Note: This example assumes that a SETROPTS GENERIC(classname) was previously issued to turn generics on for this class and that a SETROPTS REFRESH was then done.

Spool Offload Considerations (JES2 Only)

You should protect offloaded information by defining the offload data set to RACF a universal access authority of NONE. If you have security labels active, you should assign the offload data a SECLABEL of SYSHIGH to prevent unauthorized access.

Offloading Data

When you offload data from the spool to another device, JES2 copies the security information for the data to the offload job and data set headers. No validation is made of the security information written to the offload data set. JES2 calls RACF to ensure the operator starting the offload operation has sufficient authority to issue the command to start the offload.

During the offload process, JES2 calls RACF (using the WRITER class) to ensure the owner of the SYSOUT data set has at least READ access to the offload device

by checking the security information associated with the data against the device's profile in the WRITER class. The offload device profile for offload SYSOUT transmitter 1 would be:

*jesx*LOCAL.OFF1.ST

During spool offload, jobs are not checked for access to the device.

Reloading Data

When JES2 reloads the information from an offload data set, it performs any security validation necessary (similar to reading a job into the system or receiving a network SYSOUT data set) before writing the data to spool by checking the JESJOBS class for reloaded jobs and the NODES class. When RACF performs the NODES class check, if the node associated with the data is in the &RACLNDE profile, RACF accepts the data.

The following profiles for the JESINPUT class apply to spool reload:

```
OFFn.JR     for jobs
OFFn.SR     for SYSOUT
```

As with offload, JES2 calls RACF to ensure the operator starting the reload operation has sufficient authority to issue the commands.

When reloading a data set that was offloaded on this node, the name of the node must be defined in the RACFVARS profile &RACLNDE, or NODES profiles are required for NJE processing to associate user IDs with jobs or data.

How RACF Affects Jobs Dumped from and Restored to Spool (JES3 Only)

RACF performs security validation for all jobs restored to your system using the JES3 Dump Job facility.

Dumping Jobs

JES3 dumps all security information associated with each job when you use the dump job facility. However, JES3 does not perform security validation while dumping jobs.

Restoring Jobs

JES3 calls RACF to revalidate the job. RACF validates the job using the security information saved when the job was dumped and writes an SMF audit record for each restored job.

Attention

Jobs and data transported to a complex that uses different security labels may be inadvertently declassified.

Authorizing Console Access

This section discusses protecting MCS consoles, remote workstations, and JES3 consoles.

MCS Consoles

Your MVS system programmer can require operators to log on to and log off from MCS-managed consoles by specifying options in the CONSOL*xx* member of the

SYS1.PARMLIB data set. When the CONSOLE class is active and a console being used is protected by a profile in the CONSOLE class, RACF ensures that the person attempting to LOGON has the proper authority to do so.

For information about controlling access to MCS consoles, see “Protecting Consoles” on page 238.

Remote Workstations (RJP/RJE Consoles)

Your JES system programmer can require that remote workstation operators enter a password during workstation logon. This can be done through RACF or by using JES initialization statement parameters.

Note: In JES2, remote workstations are called RJE consoles. In JES3, they are called RJP consoles. If the workstation is connected using BSC, the operator must issue a /*SIGNON statement. If the workstation is connected using SNA, the operator must issue a LOGON statement.

If you want RACF to check LOGON or /*SIGNON passwords, you must activate the FACILITY class and define a profile for each workstation in both the FACILITY and USER classes. You should also ask your JES system programmer for the workstation name. If JES2 is installed, the workstation name has the form RMT*nnnn*, where *nnnn* is the remote workstation number. If JES3 is installed, the workstation name is derived from the RJPWS initialization statement for an SNA workstation or the RJPTERM initialization statement for BSC. This workstation name serves as the user ID for the workstation console. Users of the RJP console have to log on using this terminal ID and supply the same password.

You may also need similar support for NJE nodes for command and user ID authorization from the network. NJE nodes do not sign on as RJE workstations do, but rather perform the FACILITY/USERID verification as each command is issued. Also see “Authorizing the Use of Operator Commands” on page 482.

Command validation in JES is composed of two parts:
1. Validating that the originator of the command can issue the command.
2. Validating that the originator is authorized to the object of the command.

RACF control is only applied to the issuance of the command. JES continues to validate what object a particular workstation or node can affect.

Notes:

1. JES password protection or command authorization is used instead of RACF protection if any of the following conditions exist:
   - RACF is not installed.
   - No NJE node or remote workstation profile exists in the FACILITY class.
   - RACF is active, but the FACILITY class is not active.
2. If RACF is installed but not active, control returns to JES, and JES does its own password checking or command authorization.
3. Workstation operators can change their user passwords only at logon time.
4. RACF password protection replaces JES password protection for remote workstations. That is, either RACF or JES, but not both, verifies logons and passwords. Similarly, RACF command authorization across the network replaces JES NJE command authorization. That is, RACF or JES, but not both, verifies these commands.
5. The password for an RJE workstation must be changed the first time the workstation issues a LOGON or SIGNON.

6. Because the remote workstation or node name is also used as a port of entry, it needs to be defined to the JESINPUT class (if active). If it is not defined and the class is later activated, RJE signons or NJE command authorizations fail because of incorrect port of entry. For more information, see *MVS/ESA and RACF 1.9 Security Implementation Guide.*

To use RACF to check LOGON or /*SIGNON passwords, take the following steps:

1. For each remote workstation or node to be protected, ask your JES system programmer for the following:
   - The ID of the remote workstation. The ID serves as the user ID of the remote workstation. All users using a particular remote workstation must log on using this ID and supply the same password. (The password will never expire.) The ID is one of the following:
     – If JES2 is installed, the remote ID of the RJE console to be protected, which takes the form RMT*nnnn*.
     – If JES3 is installed, the ID of the console you want to protect.
     – For NJE nodes, the name of the node to be used as the user ID of that node.
2. For each remote workstation or NJE node, create a user profile:

```
ADDUSER userid
        DATA('data')
        PASSWORD(initial-password)
        DFLTGRP(groupname)
```

   where:

userid	is the RJE remote ID or NJE node name.
data	is installation-defined, for example: `DATA('RJE console at xxx, phone yyy')`
initial-password	is the initial password (to be changed immediately to another password that will never expire).
groupname	is a group that you allow to use certain RACF-protected resources, such as commands.

   Specify that the passwords for these profiles will never expire:

```
PASSWORD USER(userid) NOINTERVAL
```

3. For each workstation for which you want RACF to check the user's password, create a profile in the FACILITY class, as follows:

```
RDEFINE FACILITY RJE.workstation
```

   where *workstation* has been supplied by the JES system programmer.

   Note: The existence of a profile in the FACILITY class for a remote workstation forces the user to enter a password to be checked by RACF, rather than by JES. The specification of UACC for these profiles has no effect.
4. For each NJE node for which you want RACF to check the user's command authorization, create a profile in the FACILITY class, as follows:

```
RDEFINE FACILITY NJE.nodename
```

   where *nodename* has been supplied by the JES system programmer. The specification of UACC for these profiles has no effect.

5. Run a batch job with old and new passwords specified to set a new password (which will never expire).
6. When you are ready to start using the protection provided by the profiles you have created, activate the FACILITY class:

   ```
   SETROPTS CLASSACT(FACILITY)
   ```

7. If the class is active, define the workstation or node name to the JESINPUT class, as follows:

   ```
   RDEFINE JESINPUT workstation UACC(appropriate-access)
   RDEFINE JESINPUT nodename    UACC(appropriate-access)
   ```

   If the workstation or node name is not defined and the class is later activated, sign on or command authorization fails because of incorrect port of entry. For more information, see *MVS/ESA and RACF 1.9 Security Implementation Guide.*

JES3 Consoles

You cannot use RACF to control access to locally attached JES3 consoles. See *z/OS JES3 Initialization and Tuning Guide*.

Controlling Where Output Can Be Processed

You can use the WRITER class to control where output can be printed. For example, you can authorize or restrict the use of writers for local printers and punches, remote workstations (RJE and RJP devices), and network nodes. You can also limit which classification of data can be sent to a particular device or node. For information about how to use the WRITER class to control outbound jobs and SYSOUT for NJE, see “Authorizing Outbound Work” on page 470.

When the WRITER class is active, RACF ensures that the user is authorized to use a writer. For network devices, RACF also verifies the security of outbound data sets to ensure that the originator is authorized to send the data set to another node in a network.

To control where output can be sent, do the following:

1. Ask your JES system programmer for the following information:
   - The name of your JES system
   - If you are protecting local printers, local punches, or RJE devices, their device names
   - If you are protecting network devices, the name of the node that will ultimately receive the output

     Note: The node name as specified in the JES initialization stream.
   - The security label if you want to limit which classifications of output can be sent to a particular output destination
   - The list of users to be authorized or restricted from using a specific output destination
2. Create a profile in the WRITER class to protect each writer:

   ```
   RDEFINE WRITER profile-name UACC(appropriate-access)
   ```

   where *profile-name* has one of the following formats:
   - For local printers and punches:

     ```
     jesname.LOCAL.devicename
     ```

   - For JES2 RJE devices:

jesname.RJE.*devicename*

- For JES3 RJP devices:

  jesname.RJP.*devicename*

- For data whose destination is a node:

  jesname.NJE.*nodename*

  where *nodename* is the name of the node to ultimately receive the output.

Also, UACC can be one of the following:

NONE Allows no access

READ Allows all users to send output to the protected device or node.

3. Give the appropriate access to users and groups:

```
PERMIT profile-name CLASS(WRITER) ID(user or group)
       ACCESS(appropriate-access)
```

   where *appropriate-access* is one of the following:

   NONE Allows no access

   READ Allows the user or group to send output to the protected device or node.

4. When you are ready to start controlling access to writers based on the profiles you have defined, activate the WRITER class:

```
SETROPTS CLASSACT(WRITER)
```

   Note: If SDSF is installed on your system, WRITER profiles control which operations related to printers (such as displaying information about a printer or purging output) users can enter on SDSF panels. For complete information on creating WRITER profiles for use with SDSF, see *z/OS SDSF Operation and Customization*.

Authorizing the Use of Your Installation's Printers

You can use RACF to control who can use your installation's printers. Printers at your installation are defined to JES3 by DEVICE statements in the JES3 initialization stream. Printers are also defined in the Hardware Configuration Definition (HCD) program.

To authorize or restrict the use of your installation's printers, take the following steps:

1. Ask your JES system programmer for the following information:

   - A 4-part profile name that represents the printer. The format of the 4-part profile name is:

     sysname.dev-class.modelno.ddd

     where:

     sysname identifies the name of the system.

     dev-class specifies the type of device. For printers, you must always specify unit record (UR).

     modelno specifies the model number of the printer.

     ddd specifies the device number associated with the printer.

- The universal access authority associated with the printer. A UACC of READ indicates the printer can be allocated to all users in your installation. A UACC of NONE indicates the printer can only be allocated to the users you specify.
- A list of users and groups that have access other than the UACC. READ access allows the device to be allocated to the job submitted by the specified user.
- The security label associated with the printer (if security labels are being used).

2. Create a profile in the DEVICES class to protect each writer:

```
RDEFINE DEVICES profile-name UACC(NONE)
```

3. When you are ready to start using the protection provided by the profiles you have created, activate the DEVICES classes:

```
SETROPTS CLASSACT(DEVICES)
```

Authorizing the Use of Operator Commands

You can control which commands operators can enter at consoles. For more information, see “Administering the Use of Operator Commands” on page 276.

Commands from RJE Work Stations

To control the commands entering from RJE workstations, do the following:

1. Add a user profile for the workstation. The user ID should be the name of the remote, with parentheses removed. For example, for RMT(1), the user ID is RMT1. If you are using RACF to sign on RJE workstations, see “Remote Workstations (RJP/RJE Consoles)” on page 478. Here is a sample command:

```
ADDUSER RMT1
        DATA('RJE workstation at xxx, phone yyy')
        PASSWORD(initial-password)
        DFLTGRP(groupname)
```

2. Permit the RJE user ID to the appropriate command profiles:

```
PERMIT command-profile-name CLASS(OPERCMDS) ID(RMT1)
       ACCESS(appropriate-access)
```

3. If the OPERCMDS class is not already active, activate it:

```
SETROPTS CLASSACT(OPERCMDS)
```

Commands from NJE Nodes

To control the commands entering from NJE nodes, do the following:

1. Take the steps to define a user profile and FACILITY class profile for each node. FACILITY/USERID for NJE commands are verified as each command comes through the network. No advance sign on exists as with RJE workstations. See “Remote Workstations (RJP/RJE Consoles)” on page 478. For example, for a node named HYDEPARK:

```
ADDUSER HYDEPARK
        DATA('NJE node at xxx, phone yyy')
        PASSWORD(initial-password)
        DFLTGRP(groupname)

RDEFINE FACILITY NJE.HYDEPARK
```

2. If the NODES class is active, create a NODES profile with RUSER as the second qualifier:

```
RDEFINE NODES nodename.RUSER.userid UACC(appropriate-access)
```

where *appropriate-access* is one of the following:

NONE	Reject the command
READ	Reverify
UPDATE or higher	Pass

3. Permit the node's user ID to the command profiles the node can issue:

```
PERMIT command-profile-name CLASS(OPERCMDS) ID(HYDEPARK)
       ACCESS(appropriate-access)
```

4. If the OPERCMDS and FACILITY classes are not already active, activate them:

```
SETROPTS CLASSACT(OPERCMDS FACILITY)
```

Who Authorizes Commands When RACF Is Active

If you are using only MCS-managed consoles and enable RACF command authority checking, RACF performs all command authorization. However, if you are also using local or remote JES3 consoles, whether JES3 or RACF performs command authority checking depends on the source from which the command was entered. For a description of command authority checking when the OPERCMDS class is active, see *z/OS JES3 Initialization and Tuning Guide*.

Chapter 16. RACF and Storage Management Subsystem (SMS)

This chapter describes using RACF with the DFSMSdfp facility Storage Management Subsystem (SMS). It describes factors that administrators should consider when using RACF with SMS. A number of scenarios are used to explain these factors. Many of the names used in these scenarios are arbitrary. The names you use for user IDs, group names, and resources will differ, and the procedures you decide to follow might vary from those given as examples in this chapter.

Overview of RACF and SMS

You can use RACF to protect and control the use of SMS classes, data sets, functions, options, and commands. RACF provides the following facilities to support DFP:

- Supplied general resource classes that you can use to protect SMS classes (general resource classes are *not* the same as SMS classes)
- A DFP segment in both user and group profiles in which you can specify default information that DFP uses to determine data management and storage characteristics for data sets
- A DFP segment in data set profiles in which you can specify the owner of SMS-managed data sets protected by the profile
- Field-level access checking to provide security for fields in the DFP segment of user, group, and data set profiles

The following sections describe the details of these RACF facilities.

RACF General Resource Classes for Protecting SMS Classes

RACF provides the following general resource classes for protecting SMS management classes and SMS storage classes. (SMS data classes do not require RACF protection.)

- **MGMTCLAS.** Use this RACF resource class to protect specific SMS management classes. (Management class is the DFP construct name for a collection of attributes related to the migration and backup of data sets.)

- **STORCLAS.** Use this RACF resource class to protect specific SMS storage classes. (Storage class is the DFP construct name for attributes related to space for a data set and the device and volume on which a data set resides.)

Note: The RACF general resource classes MGMTCLAS and STORCLAS are different from, and should not be confused with, the DFP construct names management class and storage class.

Controlling the Use of SMS Classes

To control the use of SMS classes, issue RACF commands as described below.

First, issue the SETROPTS command with the CLASSACT operand to activate the RACF general resource classes MGMTCLAS and STORCLAS. The format of the command is as follows:

```
SETROPTS CLASSACT(MGMTCLAS STORCLAS)
```

Then, to define a specific SMS class, issue the RDEFINE command and specify the appropriate operands. After you define a profile to protect a specific SMS class, issue the PERMIT command to create entries in the access list of the profile. You might want to look at “Determining the Owner of an SMS-Managed Data Set” on page 490 for more information.

For example, suppose you want to define a profile in the RACF general resource class STORCLAS to protect an SMS storage class named DFP2STOR. You can control which users and groups can use DFP2STOR by issuing one of the following sequences of commands:

- To limit the number of users who can use DFP2STOR:
  1. Issue the RDEFINE command to define the profile for DFP2STOR and assign a UACC of NONE to the profile. The format of the command is as follows:

     ```
     RDEFINE STORCLAS DFP2STOR UACC(NONE)
     ```

     This command specifies that *no* users can access DFP2STOR, except for the creator of the profile. For more information, see *z/OS SecureWay Security Server RACF Command Language Reference*.
  2. Selectively *allow* certain users and groups access to DFP2STOR by issuing the PERMIT command and specifying an ACCESS of READ. The format of the command is as follows:

     ```
     PERMIT DFP2STOR CLASS(STORCLAS) ID(SMITH JONES) ACCESS(READ)
     ```

     This command allows SMITH and JONES the use of storage class DFP2STOR.
- To allow many users the use of DFP2STOR:
  1. Issue the RDEFINE command to define the profile for DFP2STOR and assign a UACC of READ to the profile. The format of the command is as follows:

     ```
     RDEFINE STORCLAS DFP2STOR UACC(READ)
     ```

     This command specifies that *all* users can access DFP2STOR.
  2. You can selectively *exclude* certain users and groups from using DFP2STOR by issuing the PERMIT command and specifying an ACCESS of NONE. The format of the command is as follows:

     ```
     PERMIT DFP2STOR CLASS(STORCLAS) ID(SMITH JONES) ACCESS(NONE)
     ```

This command prevents SMITH and JONES from using storage class DFP2STOR.

- For SMS resource classes that you want to be available to all users, consider creating an entry in the global access checking table. For example, to allow all users access to DFP2STOR, enter:

```
RDEFINE GLOBAL STORCLAS ADDMEM(DFP2STOR/READ)

SETROPTS GLOBAL(STORCLAS) REFRESH
```

  Global access checking helps reduce processing overhead associated with RACF authorization checking. For SMS resources that you want to have available to a limited number of users, consider using SETROPTS RACLIST processing for STORCLAS and MGMTCLAS to provide the best performance.

After you define profiles in the MGMTCLAS and STORCLAS resource classes, you should activate SETROPTS RACLIST processing for these classes. This can improve performance by reducing I/O to the RACF database.

To activate SETROPTS RACLIST processing for the MGMTCLAS and STORCLAS resource classes, issue the SETROPTS command with the RACLIST operand and specify the appropriate RACF resource class names. The format of the command is as follows:

```
SETROPTS RACLIST(STORCLAS MGMTCLAS)
```

For more information, see “SETROPTS RACLIST Processing” on page 131.

Refreshing Profiles for SETROPTS RACLIST Processing for MGMTCLAS and STORCLAS

If SETROPTS RACLIST processing has been activated for the MGMTCLAS and STORCLAS resource classes, you must refresh profiles for RACLIST processing for either class when you do one of the following:

- Define a new profile in the class
- Make changes to existing profiles in the class

Refreshing profiles for SETROPTS RACLIST processing for a RACF resource class ensures that the most current copy of a profile resides in storage and is available for RACF authorization checking.

To refresh profiles for SETROPTS RACLIST processing for the MGMTCLAS or STORCLAS resource classes, issue the SETROPTS command with the RACLIST and REFRESH operands and specify the appropriate RACF resource class names. The following command refreshes profiles for SETROPTS RACLIST processing for both MGMTCLAS and STORCLAS:

```
SETROPTS RACLIST(STORCLAS MGMTCLAS) REFRESH
```

For more information, see “Refreshing Profiles for SETROPTS RACLIST Processing” on page 133.

DFP Segment in RACF Profiles

To support DFP, RACF provides a DFP segment in user, group, and data set profiles. The following sections describe the information you can specify in this segment, how RACF and DFP use this information, and how you can use field-level access checking to control access to the DFP segment.

DFP Segment in User and Group Profiles

When SMS is installed and active on your system, every SMS-managed data set is assigned the following DFP constructs:

- Data class, which contains attributes related to the allocation of the data set
- Management class, which contains attributes related to the migration and backup of the data set
- Storage class, which contains attributes related to space for the data set and the device and volume on which the data set resides.

RACF provides the DFP segment in user and group profiles in which you can specify default values for these constructs as well as a data application identifier. During allocation of a new SMS-managed data set, RACF retrieves these default values for DFP. DFP, in turn, uses these values as input to the automatic class selection (ACS) routines that are used by SMS to assign constructs to the new data set.

The fields contained in the DFP segment of user and group profiles are as follows:

- DATAAPPL, which specifies the identifier for the data set application
- DATACLAS, which specifies the default data class
- MGMTCLAS, which specifies the default management class
- STORCLAS, which specifies the default storage class

For user and group profiles, you can specify information in the DFP segment using one of the following commands:

- ADDUSER, when defining a new user profile
- ALTUSER, when changing a user profile
- ADDGROUP, when defining a new group profile
- ALTGROUP, when changing a group profile

When defining or changing values in the DFP segment of user or group profiles, you should consider the following:

- The values that you specify for MGMTCLAS and STORCLAS must be defined as profiles in their respective RACF general resource classes and the user or group must be granted at least READ access. Otherwise, RACF does not allow the user or group to use the specified SMS class. For more information, see “Controlling the Use of SMS Classes” on page 486.
- RACF does not control access for DATAAPPL or DATACLAS. However, the values you specify in these fields should be defined for use on your system.
- Your storage administrator defines the names for the DFP constructs data class, management class, and storage class. To determine what construct names have been defined on your system, you can display a list of these names by using the Interactive Storage Management Facility (ISMF). For information on how to use ISMF, see *z/OS DFSMS: Using the Interactive Storage Management Facility*.

You can display the information in the DFP segment of a user profile by issuing the LISTUSER command with the DFP operand and, for a group profile, by issuing the LISTGRP command with the DFP operand. For more information on the RACF commands, see *z/OS SecureWay Security Server RACF Command Language Reference*.

Note: If you want to display the information in the DFP segment of any RACF profile, you must have the SPECIAL or AUDITOR attribute or at least READ

access to the segment through field-level access checking. For information on field-level access checking for the DFP segment, see “Controlling Access to the DFP Segment” on page 490.

Choosing Different Default Values for DFP Constructs

In most cases, the default values for constructs specified in the DFP segment of a user or group profile are sufficient for managing new data sets. When defining a new SMS-managed data set, however, a user can choose different default values for any of the following fields by using JCL or dynamic allocation:

- DATACLAS
- MGMTCLAS
- STORCLAS

For more information, see *z/OS MVS JCL User's Guide*.

DFP Segment in Data Set Profiles

In data set profiles, the DFP segment contains the RESOWNER field in which you can specify the owner (RACF-defined user or group) of an SMS-managed data set protected by the profile. When a user allocates a new SMS-managed data set protected by this profile, the user ID or group ID that you specify in the RESOWNER field must have at least READ access authority to the MGMTCLAS or STORCLAS profile used in the allocation. If RESOWNER is not specified, the user or group name matching the high-level qualifier is used. In most cases, the owner of an SMS-managed data set is the user ID or group name that matches the high-level qualifier of the data set name. RACF provides the RESOWNER field to give your installation the flexibility to select any RACF-defined user or group to be the data set owner.

You should specify a value for RESOWNER when the owner of a data set must be different from the high-level qualifier of the data set name. For example, assume that you have defined the groups PAYROLL and LEGAL on your system. Assume also that PAYROLL needs to create some data sets for LEGAL, but LEGAL requires ownership of the data sets. If you issue the following command, you create the data set profile PAYROLL.LGL88.** with LEGAL as owner of any SMS-managed data sets protected by the profile:

```
ADDSD 'PAYROLL.LGL88.**' DFP(RESOWNER(LEGAL)) UACC(NONE)
```

The PAYROLL group can then create data sets such as PAYROLL.LGL88.WEEK1, PAYROLL.LGL88.WEEK2, and PAYROLL.LGL88.MARCH.SUM, but the LEGAL group actually owns the data sets.

(The profile name PAYROLL.LGL88.** is a generic profile name that uses enhanced generic naming. Before you issue the above command, both generic profile checking for the DATASET class and enhanced generic naming must be active. If these options are not active, issue the SETROPTS GENERIC(DATASET) and SETROPTS EGN commands before you define the generic profile.)

You can specify a value for RESOWNER when you define a new data set profile using the ADDSD command or when you change an existing data set profile using the ALTDSD command. You can display the information in this field using the LISTDSD command. See *z/OS SecureWay Security Server RACF Command Language Reference* for more information on these commands.

Note that the RESOWNER field, which represents the data set owner for data set allocation purposes, is different from the OWNER field, which represents the user or group that owns the data set profile and can therefore work with the profile itself.

How RACF Uses the Information in the DFP Segments

When a user creates a new SMS-managed data set, RACF uses the information in the DFP segment of a data set profile together with the information in the DFP segment of a user or group profile as described in the following sections.

Determining the Owner of an SMS-Managed Data Set

DFP invokes RACF during data set allocationto determine the owner of the data set.

If the data set is not protected by a profile, RACF returns the high-level qualifier of the data set name as the default value for the owner of the data set.

If there is a data set profile, RACF checks it to determine whether the DFP segment contains a value for the RESOWNER field. If the RESOWNER field contains a value (user ID or group name), RACF returns this value to DFP as the owner of the data set. If the RESOWNER field does not contain a value, RACF uses the high-level qualifier of the data set name as the default value for the owner of the data set. In either situation, the value that RACF returns can be used as an input variable to ACS routines.

Retrieving Default DFP Information from User and Group Profiles

To have SMS use RACF for retrieving default values from the DFP segment of user and group profiles, your installation must specify ACSDEFAULTS=YES in the IGDSMS*mm* member of SYS1.PARMLIB. For more information, see *z/OS DFSMSdfp Storage Administration Reference*.

After invoking RACF to determine the owner (user or group) of an SMS-managed data set, DFP again invokes RACF to retrieve the default values for DATAAPPL, DATACLAS, MGMTCLAS, and STORCLAS from the DFP segment of the owner's RACF profile. DFP uses these values as input to ACS routines during allocation of the data set.

Authorization Checking for Protected SMS Classes: During allocation of an SMS-managed data set, DFP performs a RACF authorization check to verify that the data set owner is allowed to use the specified MGMTCLAS and STORCLAS. If the data set owner does not have at least READ authority to both of these classes, RACF denies the request. As a result, DFP does not allocate the data set.

This authorization checking occurs regardless of the source of the DATACLAS, MGMTCLAS, or STORCLAS values. It does not matter whether they were supplied as RACF defaults, through the ACS routines, or by the user through JCL or dynamic allocation parameters.

Note: If the data set owner is a revoked user ID, the allocation fails.

Controlling Access to the DFP Segment

You can use field-level access checking to control a user's ability to add, delete, modify, or access information in any field of the DFP segment of a user, group, or data set profile. To implement field-level access checking, issue RACF commands as described in the following sections.

Activating the FIELD Class

First, activate the FIELD general resource class (if it is not already active). To activate this class, issue the SETROPTS command with the CLASSACT operand as follows:

```
SETROPTS CLASSACT(FIELD)
```

Defining Profiles for Field-Level Access Checking

After you activate the FIELD class, you can define profiles that allow you to control access to fields in the DFP segment of user, group, or data set profiles. (Note that you can allow other users to define such profiles by assigning them the CLAUTH attribute for the FIELD class.) To define a profile for field-level access checking, issue the RDEFINE command and specify the class name as FIELD, the appropriate profile name, and the universal access authority (UACC) as follows:

```
RDEFINE FIELD profile-name UACC(access-authority)
```

When you specify *profile-name*, use the format shown in the following examples.

Controlling Access to All Fields in the DFP Segment of User Profiles: You can define a profile that lets you control access to *all* of the fields in the DFP segment of all user profiles. Before you define this profile, generic profile checking for the FIELD class must be active. If generic profile checking is not active, issue the SETROPTS GENERIC(FIELD) command.

To define this profile, issue the RDEFINE command with a generic profile name. For example, enter:

```
RDEFINE FIELD USER.DFP.* UACC(NONE)
```

Note: When you specify a UACC of NONE, you prevent all users from accessing the DFP segment in all user profiles, including their own. Likewise, if you specify a UACC of READ, you allow all users to read the information contained in all fields of the DFP segment for all user profiles.

If the FIELD class is not yet RACLISTed, you must enter the following command after you define or alter the profile:

```
SETROPTS RACLIST(FIELD)
```

If the FIELD class is already RACLISTed, you must refresh the profiles with the following command after you define or alter the profile:

```
SETROPTS RACLIST(FIELD) REFRESH
```

Controlling Access to a Specific Field in the DFP Segment of User Profiles: You can define a profile that allows you to control access to a *specific* field in the DFP segment of all user profiles by issuing the RDEFINE command and specifying *profile-name* as shown in the following example:

```
RDEFINE FIELD USER.DFP.DATACLAS UACC(NONE)
```

This command defines a profile in the FIELD general resource class that protects, with a UACC of NONE, the DATACLAS field in the DFP segment of all user profiles. For more information, see “Field-Level Access Checking” on page 213.

If the FIELD class is not yet RACLISTed, you must enter the following command after you define or alter the profile:

```
SETROPTS RACLIST(FIELD)
```

If the FIELD class is already RACLISTed, you must refresh the profiles with the following command after you define or alter the profile:

```
SETROPTS RACLIST(FIELD) REFRESH
```

Controlling Access to All Fields in the DFP Segment of Group Profiles: You can define a profile that allows you to control access to *all* fields in the DFP segment of all group profiles. Before you define the following profile, generic profile

checking for the FIELD class must be active. If it is not active, issue the SETROPTS GENERIC(FIELD) command before you define the generic profile. To define this profile, issue the RDEFINE command and specify GROUP.DFP.* for *profile-name* as shown in the following example:

```
RDEFINE FIELD GROUP.DFP.* UACC(NONE)
```

Note: When you specify a UACC of NONE, you prevent all users from accessing the DFP segment in all group profiles, including their current connect group. Likewise, if you specify a UACC of READ, you allow all users to read the information contained in all fields of the DFP segment for all group profiles.

If the FIELD class is not yet RACLISTed, you must enter the following command after you define or alter the profile:

```
SETROPTS RACLIST(FIELD)
```

If the FIELD class is already RACLISTed, you must refresh the profiles with the following command after you define or alter the profile:

```
SETROPTS RACLIST(FIELD) REFRESH
```

Controlling Access to a Specific Field in the DFP Segment of Group Profiles: You can define a profile that allows you to control access to a *specific* field in the DFP segment of all group profiles by issuing the RDEFINE command and specifying *profile-name* as shown in the following example:

```
RDEFINE FIELD GROUP.DFP.STORCLAS UACC(NONE)
```

This command defines a profile in the FIELD general resource class that protects, with a UACC of NONE, the STORCLAS field in the DFP segment of all group profiles. For more information, see “Field-Level Access Checking” on page 213.

If the FIELD class is not yet RACLISTed, you must enter the following command after you define or alter the profile:

```
SETROPTS RACLIST(FIELD)
```

If the FIELD class is already RACLISTed, you must refresh the profiles with the following command after you define or alter the profile:

```
SETROPTS RACLIST(FIELD) REFRESH
```

Controlling Access to All Fields in the DFP Segment of Data Set Profiles: You can define a profile that allows you to control access to *all* fields in the DFP segment of all data set profiles. Before you define this profile, generic profile checking for the FIELD class must be active. If generic profile checking is not active, issue the SETROPTS GENERIC(FIELD) command. To define this profile, issue the RDEFINE command and specify DATASET.DFP.* for *profile-name* as shown in the following example:

```
RDEFINE FIELD DATASET.DFP.* UACC(NONE)
```

Note: When you specify a UACC of NONE, you prevent all users from accessing the DFP segment in all data set profiles, including data set profiles that they own. Likewise, if you specify a UACC of READ, you allow all users to read the information contained in all fields of the DFP segment for all data set profiles.

If the FIELD class is not yet RACLISTed, you must enter the following command after you define or alter the profile:

```
SETROPTS RACLIST(FIELD)
```

If the FIELD class is already RACLISTed, you must refresh the profiles with the following command after you define or alter the profile:

```
SETROPTS RACLIST(FIELD) REFRESH
```

Controlling Access to a Specific Field in the DFP Segment of Data Set Profiles: You can define a profile that allows you to control access to a *specific* field in the DFP segment of all data set profiles by issuing the RDEFINE command and specifying *profile-name* as shown in the following example:

```
RDEFINE FIELD DATASET.DFP.RESOWNER UACC(NONE)
```

This command defines a profile in the FIELD general resource class that protects, with a UACC of NONE, the RESOWNER field in the DFP segment of all data set profiles. For more information, see “Field-Level Access Checking” on page 213.

If the FIELD class is not yet RACLISTed, you must enter the following command after you define or alter the profile:

```
SETROPTS RACLIST(FIELD)
```

If the FIELD class is already RACLISTed, you must refresh the profiles with the following command after you define or alter the profile:

```
SETROPTS RACLIST(FIELD) REFRESH
```

Creating the Access List for Field-Level Access Checking

After you define a profile to protect a resource in the FIELD class, you can create entries in the resource’s access list using the PERMIT command. The following example shows how to create an entry that gives user DFPADMIN the authority to alter the SMS management class (MGMTCLAS field) in the profiles of all DFP users. Note that UPDATE authority is sufficient to change a value in a field of the DFP segment.

```
PERMIT USER.DFP.MGMTCLAS CLASS(FIELD) ID(DFPADMIN) ACCESS(UPDATE)
```

You can also specify the value &RACUID with the ID operand on the PERMIT command. When you enter this value on the PERMIT command, you allow all users access to the specified field within the DFP segment of their own user profiles. For example, if you issue the following command, you allow all users to read the DATAAPPL field in the DFP segment of their own user profiles.

```
PERMIT USER.DFP.DATAAPPL CLASS(FIELD) ID(&RACUID) ACCESS(READ)
```

Controlling the Use of Other SMS Resources

You can use RACF to control access to the following SMS resources:

- The Interactive Storage Management Facility (ISMF), including:
  - The entire ISMF component
  - Individual ISMF applications
  - ISMF functions, line operators, and commands
- The execution of functions, options, and commands, including:
  - Catalog functions for SMS data sets
  - DFSMSdss functions on data sets
  - Using SMS to activate a configuration
- SMS data sets, including:
  - Control data sets
  - Source data sets for ACS routines
  - The test library for ACS routines

For information on how to use RACF to protect these resources, see the following publications:

- *MVS/ESA SML: Managing Data*, which shows the sequence of RACF commands you need to issue to protect the various SMS resources
- *z/OS DFSMSdfp Storage Administration Reference* and *z/OS DFSMS: Managing Catalogs*, which show the names of the profiles you need to define to protect the various SMS resources.

Chapter 17. RACF and TSO/E

This chapter describes using RACF with TSO/E.

TSO/E Administration Considerations

In order for users to log on to TSO, they must have an entry in the SYS1.UADS data set or a TSO segment defined in their RACF user profile. For more information, see “The TSO Segment in User Profiles” on page 69.

Note: A TSO installation can write a TSO logon pre-prompt exit to bypass checking SYS1.UADS for user attribute information. For more information, see *z/OS TSO/E Customization*.

You can move TSO user attribute information from SYS1.UADS to the RACF database. (SYS1.UADS contains an entry for each TSO user that describes the attributes that regulate the user’s access to the system.) When you move this TSO information into the RACF database, it is stored in the TSO segment of the user’s profile. When a user logs on to TSO, it uses the information contained in the TSO segment to build a session for the user.

Moving the TSO user information to the RACF database eliminates the need to maintain an entry in SYS1.UADS for each TSO user. However, you *must* maintain entries in SYS1.UADS for certain users (such as IBMUSER and system programmers).

For example, if you need to deactivate RACF to perform maintenance on the RACF database, users authorized to perform this maintenance must be able to log on to the system. When RACF is inactive, TSO checks entries in SYS1.UADS to authorize access to the system. When RACF is active, logon verification can produce an error during RACF processing. However, the logon can proceed by an alternative method (for example, UADS). This error occurs if the installation does not use the RACF database to store security-related information for a particular user, but it does use an alternative method (such as UADS) for the logon application to perform user verification.

Note: You can use the RACONVRT EXEC to help you convert SYS1.UADS entries to RACF user profiles. The RACONVRT EXEC creates a CLIST that contains multiple members. Each member contains RACF commands needed to add information read from the SYS1.UADS data set to the RACF database.

Be sure to inspect *all* members before running them. In particular, all of the ADDUSER commands that RACONVRT generates connect the users to group SYS1. *Be sure to modify the default groups before running this member.* You should

also check the completeness and accuracy of the conversion that is performed by the RACONVRT EXEC. For more information on using the RACONVRT EXEC, see *z/OS TSO/E Customization*.

Protecting TSO Resources

You can use RACF to protect certain TSO resources. These resources include TSO logon procedures, account numbers, and performance groups. In addition, you can protect resources called TSO user authorities, whose settings determine whether a user can issue certain authorized TSO commands. Examples of TSO user authorities include ACCT, JCL, MOUNT, OPER, RECOVER, PARMLIB, TESTAUTH, and CONSOLE. For detailed information about the TSO resources you can protect with RACF, see *z/OS TSO/E Customization*.

If you are defining TSO segments in user profiles, you must protect these TSO resources, using the following general resource classes:

- TSOPROC (for protecting TSO logon procedures)
- ACCTNUM (for protecting TSO account numbers)
- PERFGRP (for protecting TSO performance groups)
- TSOAUTH (for protecting TSO user authorities)

The following access authorities apply to these resources:

NONE No access allowed.

READ For TSOPROC, ACCTNUM, and PERFGRP, allows users to specify the logon procedure, account number, or performance group when logging on.

For TSOAUTH, gives the user the authority to issue the associated authorized TSO command.

For PARMLIB, allows the user to issue the PARMLIB LIST command.

For TESTAUTH, allows the user to invoke a program in authorized state.

UPDATE For PARMLIB, allows the user to issue the PARMLIB UPDATE command. For the other profiles, UPDATE is the same as READ.

CONTROL Same as READ.

ALTER Allows users to change the profile, if the profile is discrete.

To control the use of TSO resources, issue RACF commands in the following sequence:

1. Activate the TSO general resource classes:

```
SETROPTS CLASSACT(TSOPROC ACCTNUM PERFGRP TSOAUTH)
```

Considerations When Activating the TSO Resource Classes

Assume that you have defined a user profile for user SMITH that contains a TSO segment.

- If you do not activate the TSOPROC and ACCTNUM classes, user SMITH cannot log on to TSO because RACF cannot check SMITH's authority to use the logon procedure and account number specified on the logon panel. TSOPROC and ACCTNUM *must* be active so that users whose profiles contain TSO segments can log on to TSO.
- If you do not activate the PERFGRP class and user SMITH specifies a performance group on the logon panel, SMITH cannot log on to TSO because RACF cannot check SMITH's authority to access the specified performance group. However, SMITH can log on to TSO when the performance group is deleted from the logon panel. Activate the PERFGRP class if your installation intends to use TSO performance groups.
- If you do not activate the TSOAUTH class, user SMITH can log on to TSO but will not have any assigned TSO user authorities such as JCL or MOUNT. Activate the TSOAUTH class and give SMITH READ access authority to the appropriate resources in the TSOAUTH class if your installation is specifying user authorities when defining users to the system.

2. Create profiles to protect TSO resources. The following example shows how to define logon procedure LOGPROC1 to the TSOPROC resource class and assign it a UACC of READ. (A UACC of READ grants all users the ability to use the logon procedure.)

   ```
   RDEFINE TSOPROC LOGPROC1 UACC(READ)
   ```

   To protect a TSO resource so that a limited number of users can access it, you can define it and specify a UACC of NONE. Then you can create an access list containing only those users who require access to the resource. The following example shows how to define a logon procedure, LOGPROC2, in the TSOPROC resource class and protect it with a UACC of NONE.

   ```
   RDEFINE TSOPROC LOGPROC2 UACC(NONE)
   ```

Considerations for Creating Profiles for TSO Resources

- For the TSOPROC class, the profile name must be the name of the logon procedure itself (no generic characters are allowed).
- For the ACCTNUM class, the profile name can be up to 39 characters long.

  You should create at least one profile in the ACCTNUM class.

  If you want a particular user to log on without an account number, you must ensure that the user has no access to any ACCTNUM profile. This means that you cannot specify UACC(READ) for *any* ACCTNUM profile. Also, a user can have access to an ACCTNUM profile by means of a connect group. If a user has access to one or more account numbers, the first such account number that RACF encounters when searching the RACF database becomes that user's default account number and is saved in the TSO segment of the user's profile. You can find out which account number is used by issuing the following command:

  ```
  SEARCH CLASS(ACCTNUM) USER(userid)
  ```

  The first account number listed is used. For example, you if you want to allow only two account numbers, D1001 and D1002, and you want to ensure that users log on with at least one of them, create the following profiles:

  ```
  RDEFINE ACCTNUM D1001 UACC(READ)
  RDEFINE ACCTNUM D1002 UACC(READ)
  RDEFINE ACCTNUM ** UACC(NONE)
  ```

  Note: Because of the order in which RACF searches the RACF database, account number D1001 is the default assigned to any user who logs on with a blank account number. To determine the search order in which profiles are used, issue SEARCH or RLIST command for the class. For example:

  ```
  SEARCH CLASS(ACCTNUM)
  ```

- For the PERFGRP class, the profile name must be the number of the performance group itself (no generic characters are allowed).
- For the TSOAUTH class, you should consider creating discrete profiles for each TSO attribute. The following examples assume that only a few users should be able to request mounts, but that every user (except those specifically disallowed) should be able to submit batch jobs:

  ```
  RDEFINE TSOAUTH MOUNT UACC(NONE)
  RDEFINE TSOAUTH JCL UACC(READ)
  ```

3. Use the PERMIT command to allow users and groups to use the TSO resources. The following example shows how to allow users USERA and USERB to specify logon procedure LOGPROC2 when they log on using TSO:

   ```
   PERMIT LOGPROC2 CLASS(TSOPROC) ID(USERA USERB) ACCESS(READ)
   ```

4. Activate SETROPTS RACLIST processing for the TSO general resource classes:

   ```
   SETROPTS RACLIST(TSOPROC ACCTNUM PERFGRP TSOAUTH)
   ```

   For more information on SETROPTS RACLIST processing, see section "SETROPTS Options to Activate In-Storage Profile Processing" on page 130.

Note: If SETROPTS RACLIST processing is already activated for the TSO general resource classes, you must refresh SETROPTS RACLIST processing:

```
SETROPTS RACLIST(TSOPROC ACCTNUM PERFGRP TSOAUTH) REFRESH
```

For more information on refreshing SETROPTS RACLIST processing, see section “Refreshing Profiles for SETROPTS RACLIST Processing” on page 133.

Authorization Checking for Protected TSO Resources

When a user logs on to TSO, TSO requests RACF authorization checking for protected TSO resources such as account numbers and logon procedures. For example, suppose that during logon, user SMITH requests the use of account number 12345. If SMITH is authorized to use account number 12345, RACF grants the request. If SMITH is not authorized to use account number 12345, the following occur:

- A message is sent to the operator console indicating that user SMITH has been denied access to a RACF-protected resource.
- An SMF record is generated indicating that RACF failed an attempt to access a protected resource (unless your installation has specified an alternative auditing option for account numbers).
- User SMITH is prompted to enter a valid account number.

RACF performs authorization checking in this manner for protected TSO resources in the TSOPROC, ACCTNUM, and PERFGRP classes. For resources in the TSOAUTH class, RACF performs authorization checking but no messages are sent to the operator console and no SMF records are generated.

Field-Level Access Checking for TSO

You can use RACF to control which users can access fields in the TSO segments of RACF profiles through *field-level access checking*. For more information on field-level access checking, see “Field-Level Access Checking” on page 213.

Controlling the Use of the TSO SEND Command

You can control whether users can receive messages sent with the TSO SEND command.

Notes:

1. When the SMESSAGE class is active and a profile does not exist for the specified user, the SEND command completes normally.
2. When the SECLABEL class is active, the receiver of the message must pass the security label authorization check based on the receiver’s current security label and the security label of the message (which was set by the sender’s current security label at the time that the sender issued the TSO SEND command.)

To control the use of the TSO SEND command, do the following:

1. Create profiles in the SMESSAGE class:

   ```
   RDEFINE SMESSAGE userid-of-receiver UACC(NONE)
   ```

2. Give users the appropriate access authority:

   ```
   PERMIT RECVR1 CLASS(SMESSAGE) ID(SENDER1) ACCESS(READ)
   PERMIT RECVR2 CLASS(SMESSAGE) ID(SENDER2) ACCESS(NONE)
   ```

where SENDER1 and SENDER2 are valid RACF user IDs or group names, and RECVR1 and RECVR2 are valid RACF user IDs. The *first* PERMIT in the example allows SENDER1 to send messages to RECVR1. The *second* PERMIT in the example prevents SENDER2 from sending messages to RECVR2.

3. When you are ready to start using the security provided by these profiles, activate both the DIRAUTH class and the SMESSAGE class, and activate SETROPTS RACLIST processing for the SMESSAGE class. SETROPTS RACLIST processing helps ensure high performance when access authorities are checked. You can do these actions in one command:

   ```
   SETROPTS CLASSACT(SMESSAGE DIRAUTH) RACLIST(SMESSAGE)
   ```

   Note: Any time you make a change to an SMESSAGE profile, you must also refresh SETROPTS RACLIST processing for the SMESSAGE class for the change to take effect. For example:

   ```
   SETROPTS RACLIST(SMESSAGE) REFRESH
   ```

Restricting Spool Access by TSO Users

Activating the JESSPOOL class provides protection against unauthorized spool access through the use of the TSO OUTPUT and RECEIVE commands, as shown in Table 36.

Only authorized users (that is, the creator of a data set or those granted access through JESSPOOL profiles) have the authority to access SYSOUT data sets when using the TSO OUTPUT command.

To allow users to use the TSO OUTPUT command, create profiles in the JESSPOOL class and grant users access that they need. For more information, see “Defining Profiles for SYSIN and SYSOUT Data Sets” on page 471.

Table 36. TSO Command Usage When RACF Protection Is Enabled

TSO Command	RACF Protection Provided
OUTPUT	Prevents unauthorized users from reading data sets or changing the attributes of data sets on spool.
RECEIVE	Prevents unauthorized users from receiving transmitted data sets that are classified at a level higher than the receiver’s current authority.

TSO Commands That Relate to RACF

This section summarizes the TSO commands that relate to RACF. To check what your system configuration should be, see “B1 Configuration Requirements” on page 12.

For complete information on logging on to TSO and on TSO commands, see *z/OS TSO/E User’s Guide* and *z/OS TSO/E System Programming Command Reference*.

- Fields on the TSO/E logon panel
  - PASSWORD and NEW PASSWORD fields: Users should keep their passwords in confidence. Users should also know how to change passwords when logging on, and should know how to specify new passwords that pass the syntax rules.

  - GROUP IDENT field: Specify this field only if list-of-groups processing is *not* in effect and if the user wants the job to run with a group other than the user's default group.
  - SECLABEL field: Specify this field to log on with a security label other than the user's current security label.
  - Users with the OIDCARD attribute must supply a valid operator identification card during logon.
- Operands on the TSO LOGON command:
  - NEWPASSWORD (to specify a new password to replace the current password)
  - GROUP (to specify the group name to which the user is connected during the terminal session).
- Settings on the TSO PROFILE command:
  - PREFIX (used in determining the RRSFLIST data set name)
  - INTERCOM (determines whether RRSF uses SEND to send messages to the command issuer for directed commands).
- Sending and receiving messages: Depending on how security labels are used on your system, and on how SETROPTS options are set, users may need to be aware of their current security label when they send or receive messages from other users. For more information, see "Controlling the Use of the TSO SEND Command" on page 499.
- Submitting and cancelling jobs: Users have flexibility with regard to which jobs they can work with using the TSO SUBMIT and CANCEL commands. For more information, see "Surrogate Job Submission" on page 451 and "Controlling Who Can Cancel Jobs by Job Name" on page 450.

Using TSO When RACF Is Deactivated

While RACF is deactivated as a result of issuing the RVARY command, only users defined in the SYS1.UADS data set can still log on to TSO, but RACF does not do any processing. (For example, a user defined in SYS1.UADS who has the ADSP attribute can allocate DASD data sets, but RACF is not called to define discrete profiles for those data sets.)

If the TSO user has the WTPMSG option active, the user receives messages in the session that indicate the identity of the resource being defined or accessed.

Therefore, you should notify users who have the ADSP attribute when RACF is not active so that they are aware that profiles for newly-created data sets are not being defined to RACF.

When RACF is reactivated, you should advise RACF users to log off TSO and log on again to ensure they are connected to their proper RACF group.

Chapter 18. RACF and z/OS UNIX

This chapter describes using RACF with z/OS UNIX. The information presented here can also be used with OS/390 UNIX unless otherwise indicated. This chapter describes factors to consider when using RACF to manage group identifiers (GIDs) and user identifiers (UIDs). It also describes how to map GIDs and UIDs to RACF group names and user IDs.

The z/OS UNIX security functions provided by RACF include user validation, file access checking, privileged user checking, and user limit checking. z/OS UNIX users are defined with RACF commands. When a job starts or a user logs on, the user ID and password are verified by RACF. When an address space requests an z/OS UNIX function for the first time, RACF:

1. Verifies that the user is defined as a z/OS UNIX user.
2. Verifies that the user's current connect group is defined as a z/OS UNIX group.
3. Initializes the control blocks needed for subsequent security checks.

Additional Reading

See *z/OS UNIX System Services Planning* for complete information on setting up and using RACF in the z/OS UNIX environment.

See "Special Considerations for Auditing z/OS UNIX" on page 119 for information on auditing z/OS UNIX security events.

See *z/OS SecureWay Security Server RACF Auditor's Guide* for complete information on auditing in the RACF environment.

Note: RACF program control does not control programs that are executed in any way that bypasses MVS contents supervision, such as load modules contained in z/OS UNIX files. Therefore, loading a program from a z/OS UNIX file prevents you from opening a data set in a PADS (program-access-to-data-sets) environment, and prevents you from loading a program from an MVS library if you only have EXECUTE authority. You should use program control to restrict access to any programs, such as these, that provide facilities for bypassing MVS contents supervision. For more information, see "Protecting Programs and Using Them with PADS" on page 253.

Defining Group Identifiers (GIDs)

You can assign a group identifier (GID) to a RACF group by specifying a GID value in the OMVS segment of the RACF group profile. When a GID is assigned to a group, all users connected to this group as their default group who have a user identifier (UID) in their user profile can use z/OS UNIX functions and can access z/OS UNIX files based on the GID and UID values assigned. If a user's current connect group is not their default group, a GID must also be assigned to the current connect group.

For example, the following command defines a GID for an existing RACF group:

```
ALTGROUP SECADMIN OMVS(GID(336))
```

For more information, see "The OMVS Segment in Group Profiles" on page 53 and "Using Default OMVS Segments in USER and GROUP Profiles" on page 506.

Although the same GID can be assigned to multiple RACF groups, it is not recommended. If you assign the same GID to multiple groups, control at an individual group level is lost because the GID is used in z/OS UNIX security checks. RACF groups that have the same GID assignment are treated as a single group during z/OS UNIX security checks.

For special considerations about using the RACF list-of-groups checking (GRPLIST) option for access to hierarchical file system (HFS) files and directories, see "GRPLIST Considerations for z/OS UNIX" on page 117.

Defining User Identifiers (UIDs)

You can assign a user identifier (UID) to a RACF user by specifying a UID value in the OMVS segment of the RACF user profile. When assigning a UID to a user, make sure that the user's default group has an assigned GID. If the user specifies a group during logon or on a batch job, this current connect group must also have an assigned GID. A user with a UID and a default group (and current connect group, if applicable) with a GID can use z/OS UNIX functions and access z/OS UNIX files based on the assigned UID and GID values. If a UID and GID are not available as described, the user cannot use z/OS UNIX functions.

For example, the following command defines a UID, and other OMVS segment information, for an existing RACF user:

```
ALTUSER KAMAL OMVS(UID(122649) HOME('/') PROGRAM('/bin/sh'))
```

For more information, see "The OMVS Segment in User Profiles" on page 68 and "Using Default OMVS Segments in USER and GROUP Profiles" on page 506.

Although you can assign the same UID to multiple users, it is not recommended. However, it may be necessary for some cases, such as superusers. If you assign the same UID to multiple users, control at an individual user level is lost because the UID is used in z/OS UNIX security checks. Users with the same UID assignment are treated as a single user during z/OS UNIX security checks.

Listing UIDs and GIDs

You can list the RACF users and groups associated with UIDs and GIDs using the following methods:

1. ISPF shell. See *z/OS UNIX System Services User's Guide* for information about using the ISPF shell.
2. RACF database unload utility (IRRDBU00). See "Using the RACF Database Unload Utility (IRRDBU00)" on page 322 for information.

If you use UNIXMAP profiles to associate RACF users and groups with UIDs and GIDs, you can also use RLIST command. For example:

- To see the RACF groups that are associated with GID 237, enter:

  ```
  RLIST UNIXMAP G237 ALL
  ```

- To see the RACF user IDs that are associated with UID 0, enter:

  ```
  RLIST UNIXMAP U0 ALL
  ```

- To see all RACF groups and user IDs associated with GIDs and UIDs, enter:

  ```
  RLIST UNIXMAP * ALL
  ```

For information about the UNIXMAP class, see "Using the UNIXMAP Class and Virtual Lookaside Facility (VLF)" on page 508.

Superuser Authority

A UID of 0 is used to define an z/OS UNIX superuser. A superuser can perform any z/OS UNIX function and passes all z/OS UNIX security checks. Users running with the trusted or privileged attribute (for example, started tasks or jobs assigned by a RACROUTE REQUEST=VERIFY exit) are considered z/OS UNIX superusers even if their assigned UID is a value other than 0.

You may choose to assign the UID of 0 to multiple RACF user IDs. However, you should seek to minimize the assignment of superuser authority in your installation. You can accomplish this by setting z/OS UNIX user limits, and by managing superuser privileges through UNIXPRIV profiles. See "Using UNIXPRIV Class Profiles to Manage z/OS UNIX Privileges" on page 511 for more information.

Setting z/OS UNIX User Limits

You can control the amount of resources consumed by certain z/OS UNIX users by setting individual limits for these users. The resource limits for the majority of z/OS UNIX users are specified in the BPXPRM*xx* member of PARMLIB. These limits apply to all users except those with UID 0 (superuser authority). Rather than assigning superuser authority to application servers and other users so they can exceed BPXPRM*xx* limits, you can individually set higher limits to these users. Setting user limits allows you to minimize the number of assignments of superuser authority at your installation and reduces your security risk.

You can specify z/OS UNIX user limits by choosing options on the ADDUSER or ALTUSER commands. The limits are stored in the OMVS segment of the user profile. The following limits may be set in the OMVS user segment:

CPUTIMEMAX Maximum CPU time (`RLIMIT_CPU`)

ASSIZEMAX	Maximum address space size (RLIMIT_AS)
FILEPROCMAX	Maximum number of files per process
PROCUSERMAX	Maximum number of processes per UID
THREADSMAX	Maximum number of threads per process
MMAPAREAMAX	Maximum memory map size

Once you have set individual user limits for users who require higher resource limits, you should consider removing their superuser authority. You should also re-evaluate your installation's BPXPRM*xx* limits and consider reducing these limits. See *z/OS UNIX System Services Planning* for more information.

Protected User IDs

You can define *protected* user IDs for started procedures associated with z/OS UNIX, such as the kernel, the initialization started procedure, and important daemons that are critical to the availability of your z/OS UNIX system. This prevents these user IDs from being revoked through inadvertent or malicious incorrect password attempts, or from being used for other purposes, such as logging on to the system. For more information, see "Defining Protected User IDs" on page 86.

Using Default OMVS Segments in USER and GROUP Profiles

You should define a UID for each user and a GID for each group that needs access to z/OS UNIX functions and resources. If you have a large number of users who need access to z/OS UNIX applications, such as FTP, you can set up RACF so that it automatically uses default OMVS segments for users and groups that do not have OMVS segments in their USER or GROUP profiles. To do this, you need to:

1. Create a FACILITY class profile called BPX.DEFAULT.USER.
2. Specify a user ID, or a user ID and group name, in the application data field of that profile.
3. Ensure that a USER profile exists for the user ID you specified in step 2, and that a UID is specified in its OMVS segment. Similarly, ensure that a GROUP profile exists for the group name, and that a GID is specified in its OMVS segment, if you specified a group name in step 2.

   RACF command processing does not perform any checking to ensure that the application data points to a valid user ID, or a valid user ID and group name, or that the USER and GROUP profiles contain OMVS segments. However, in order for default OMVS segment processing to occur for users who do not have individual OMVS segments, the USER or GROUP profile must exist.

z/OS UNIX user limits specified in the OMVS segment of the default USER profile are used during default OMVS segment processing. Therefore, if you expect a large number of users to use the default USER OMVS segment, you might consider setting user limits there rather than raising system limits.

If *any* of the following are true, default OMVS segment processing will not occur:

- The FACILITY class is inactive.
- The FACILITY class profile BPX.DEFAULT.USER does not exist.
- There is no application data in the BPX.DEFAULT.USER profile.
- The application data in the BPX.DEFAULT.USER profile does not contain the name of an existing USER profile, or the names of existing USER and GROUP profiles.

- When looking for a UID, the USER profile found using the application data does not contain an OMVS segment, or its OMVS segment does not contain a UID.
- When looking for a GID, the GROUP profile found using the application data does not contain an OMVS segment, or its OMVS segment does not contain a GID.

The processing of the default OMVS segments for the user and the current connect group are independent of each other. The OMVS segment of the user specified on the `initUSP` callable service may be used to obtain the UID, and the GID may come from the group name specified in the FACILITY class profile. Similarly, when the default UID found via the user ID specified in the FACILITY class profile is used, the GID may come from the user's current connect group. Also, the user ID specified in the FACILITY class profile does not need to be a member of the group name specified in that profile. These values are used independently.

In the following example, the security administrator sets up the default USER and GROUP OMVS segments in preparation for the installation's migration to z/OS UNIX. In this case, a new user and group profile will be created to contain the defaults, and the FACILITY class has already been activated.

```
ADDUSER  OEDFLT NAME('OE DEFAULT USER') OMVS(UID(888888) HOME(/u/OEDFLT))
ADDGROUP OEGROUP OMVS(GID(17))
RDEFINE  FACILITY BPX.DEFAULT.USER APPLDATA('OEDFLT/OEGROUP')
```

The format of the application data is exactly as shown when a default is being set up for both USER and GROUP OMVS segments. To set up a default for the USER OMVS segment only, the format is:

```
APPLDATA('OEDFLT')
```

Note: When defining APPLDATA in the BPX.DEFAULT.USER profile, you may specify a user ID from which to obtain default OMVS segment information, or you may specify a user ID and a group name, but you cannot specify only a group name.

If you wish to use individual USER OMVS segments but make use of a default GROUP OMVS segment, you must specify both a valid user ID and a valid group name in the APPLDATA field. RACF will ignore the default USER OMVS specification for any user who has an individual OMVS segment. Similarly, RACF will ignore the default GROUP OMVS specification for any group that has an individual OMVS segment.

At your option, you may define individual USER OMVS segments that do not contain UIDs for certain user IDs. This will prevent these user IDs from being used to access z/OS UNIX services. For example, daemons will be unable to switch to these user IDs.

As an alternative, you may wish to use the default GROUP OMVS segment but prevent most users at your installation from accessing z/OS UNIX services. Do this by defining a USER OMVS segment that does not contain a UID in the USER profile specified in the APPLDATA field of the BPX.DEFAULT.USER profile. RACF will ignore the default USER OMVS specification for any users who have individual OMVS segments. If these users have UIDs defined in their OMVS segments, they will be able to access z/OS UNIX services.

z/OS UNIX Performance Considerations

Associating RACF user IDs and groups to UIDs and GIDs has important performance considerations. If your installation shares the RACF database with systems running releases prior to OS/390 Version 2 Release 10, it is important to use the Virtual Lookaside Facility (VLF) classes IRRUMAP and IRRGMAP and the UNIXMAP class to improve performance by avoiding sequential searches of the RACF database for UID and GID associations. If your installation shares the RACF database with only systems running z/OS, or OS/390 Version 2 Release 10 or above, you may be able to achieve improved performance without using UNIXMAP and VLF. However, before you can avoid using UNIXMAP and VLF, you must ask your system programmer if your installation has reached stage 3 of application identity mapping by running the IRRIRA00 conversion utility. If your installation is new to RACF and you are not running any releases prior to OS/390 Version 2 Release 10, you will automatically take advantage of application identity mapping at the stage 3 level without running the IRRIRA00 conversion utility, and you will not need to use VLF and UNIXMAP to achieve improved performance.

Converting to Stage 3 of Application Identity Mapping

Your system programmer can convert your RACF database to stage 3 of application identity mapping using the IRRIRA00 conversion utility. See *z/OS SecureWay Security Server RACF System Programmer's Guide* for information about running the IRRIRA00 conversion utility. In stage 3, RACF locates application identities, such as UIDs and GIDs, for users and groups by using an alias index that is automatically maintained by RACF. This allows RACF to more efficiently handle authentication and authorization requests from applications such as z/OS UNIX than was possible using other methods, such as the UNIXMAP class and VLF. Once your installation reaches stage 3 of application identity mapping, you will no longer have UNIXMAP class profiles on your system, and you can deactivate the UNIXMAP class and remove VLF classes IRRUMAP and IRRGMAP.

Using the UNIXMAP Class and Virtual Lookaside Facility (VLF)

If your installation shares the RACF database with systems running releases prior to OS/390 Version 2 Release 10, it is important to use Virtual Lookaside Facility (VLF) and the UNIXMAP class to improve performance. You may also need to use VLF and UNIXMAP class if your system programmer has not yet converted your systems for stage 3 of application identity mapping. See *z/OS SecureWay Security Server RACF System Programmer's Guide* for information about converting to stage 3 by running the IRRIRA00 conversion utility.

VLF (and the associated VLF classes IRRUMAP and IRRGMAP) and the UNIXMAP class are used to map UIDs to RACF user IDs and GIDs to RACF group names. Both VLF and the UNIXMAP class can be either active or inactive. Table 37 shows how these states affect performance. It is recommended that both the UNIXMAP class and VLF remain active, and that the VLF classes IRRUMAP and IRRGMAP should be defined to VLF.

Table 37. The UNIXMAP Class and VLF: Effects on Performance for Installations That Have Not Reached Stage 3 of Application Identity Mapping

State		Performance
Active	UNIXMAP class	Running in this state at all times will give you the best performance.
Active	VLF	

Table 37. The UNIXMAP Class and VLF: Effects on Performance for Installations That Have Not Reached Stage 3 of Application Identity Mapping (continued)

State		Performance
Active **Inactive**	UNIXMAP class VLF	If VLF is inactive, requests for UID-to-user-ID mapping and GID-to-group-name mapping must access a UNIXMAP class profile in the database, which degrades performance. Running with VLF inactive should be done only when you need to stop VLF to make changes to it.
Inactive **Active**	UNIXMAP class VLF	If the UNIXMAP class is inactive, requests for UID-to-user-ID mapping and GID-to-group-name mapping must search the entire RACF database when the UID or GID specified is not found in VLF. Running in this state degrades performance severely. The inactive state for the UNIXMAP class is provided as a migration aid. After migration is complete, you should never need to run with the UNIXMAP class inactive.
Inactive **Inactive**	UNIXMAP class VLF	Running with both VLF inactive and the UNIXMAP class inactive causes requests for UID-to-user-ID mapping and GID-to-group-name mapping to default to searching the RACF database on each request. Running in this state significantly degrades performance of these functions. It could also affect other systems in a complex sharing the RACF database because of the increased I/O to the database. It is recommended that you never run in this state.

You have the option to cache additional z/OS UNIX security information in VLF. This capability allows RACF to avoid accessing the RACF database when called to create a security environment for z/OS UNIX users. To use the cached user security (USP) packet, the IRRSMAP class must be defined to VLF. For more information, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

For information about the effect of certain RACF commands on VLF, see "RACF Commands for Flushing a VLF Cache" on page 304.

For more information on VLF, see:

- *z/OS MVS Planning: Operations*
- *z/OS MVS Initialization and Tuning Guide*
- *z/OS MVS Initialization and Tuning Reference*

Activating the UNIXMAP Class

The UNIXMAP class is not used for UID and GID lookups until you activate it at your installation. It should be left inactive until the steps listed below are performed to initially populate the UNIXMAP class with information that may already exist in user and group profiles in the database having OMVS segments. z/OS UNIX can be active while the initial population takes place. Once the UNIXMAP class is initially populated, it should be activated.

How to Initially Populate the UNIXMAP Class: If your installation already uses z/OS UNIX, or OS/390 UNIX, and has OMVS segments defined in group or user profiles, you should perform the following steps. If you have not used z/OS UNIX or OS/390 UNIX, you do not need to perform these steps.

To initially populate the UNIXMAP class, do the following:

1. Quiesce administrative activity against users and groups.
2. Run the database unload utility (IRRDBU00) against the database.
3. Read instructions at the beginning of the REXX migration exec (in the IRR30858 member of SYS1.SAMPLIB) concerning what data sets are to be used in your environment. After reading the exec and modifying it appropriately, run it against the database unload utility output. It produces a file containing RDEFINE and PERMIT commands that will populate the UNIXMAP class. Do not execute this command file yet.
4. Issue SETROPTS NOADDCREATOR. This is very important because you do not want the ID of the user who runs the command file produced in step 3 on the access list of all the profiles in this new class.
5. Execute the command file produced in step 3. When you execute this file, you may see messages ICH408I and ICH10102I, indicating that some profile is already defined to the UNIXMAP class. This occurs if a UID maps to more than one user ID or if a GID maps to more than one group name.
6. If SETROPTS ADDCREATOR was in effect prior to step 4, issue SETR ADDCREATOR now to restore that setting.
7. Issue SETROPTS CLASSACT(UNIXMAP). The UNIXMAP profiles will now be used to do UID and GID lookups. To maintain performance, it is recommended that the UNIXMAP class remain active.

Administrative activity can now be resumed against users and groups. From this point, RACF automatically keeps the UNIXMAP profiles synchronized with the user and group profiles.

Using UNIXMAP Class Profiles to Map UIDs and GIDs

For each UID that is defined in the OMVS segment of a USER profile, a general resource profile named U*uid* in the UNIXMAP class is automatically created. The access list of the U*uid* profile contains all user IDs that have been assigned this UID.

For each GID that is defined in the OMVS segment of a GROUP profile, a general resource profile named G*gid* in the UNIXMAP class is automatically created. The access list of the G*gid* profile contains all groups that have been assigned this GID.

These mapping profiles are used to provide a cross reference to USER and GROUP profiles. They provide RACF with a performance-sensitive method of returning information for a given UID or GID when requested by z/OS UNIX or application programs.

RACF automatically maintains these mapping profiles when UIDs and GIDs are added, changed, or deleted. The UNIXMAP class does not have to be active for this to happen. RACF does this by modifying UNIXMAP class profiles appropriately when ADDUSER, ALTUSER, DELUSER, ADDGROUP, ALTGROUP, or DELGROUP commands are issued. When RACF creates UNIXMAP profiles as a result of an ADDUSER, ALTUSER, ADDGROUP or ALTGROUP command, the user ID of the command issuer becomes the owner of the UNIXMAP profile.

For example, if the following command is issued:

```
ADDUSER ORTIZ OMVS(UID(13))
```

RACF creates a UNIXMAP profile named U13 with ORTIZ contained on the access list. If the following command is subsequently issued:

```
ALTUSER ORTIZ OMVS(UID(55))
```

RACF deletes the U13 profile and creates a U55 profile with ORTIZ contained on the access list.

In general, you should not alter these profiles. However, it is possible that they might get inadvertently deleted, or damaged by database corruption. If a profile is deleted, or if the user is not contained in its access list, RACF will not be able to retrieve information for the UID or GID that the profile represented. RACF will be unable to locate the mapping profile and will send z/OS UNIX a return code indicating that the UID or GID is not known.

If this happens, an authorized user needs to repair the damage. First, see if the user name associated with the UID or the group name associated with the GID can be determined from a message displayed by RACF. For example, suppose you received an error message associated with user ORTIZ. You should display the UID associated with the user profile for ORTIZ by entering:

```
LISTUSER ORTIZ OMVS NORACF
```

If, for example, LISTUSER displays a UID of 13, you would then enter:

```
RDEFINE UNIXMAP U13 UACC(NONE)
PERMIT U13 CLASS(UNIXMAP) ACCESS(NONE) ID(ORTIZ)
PERMIT U13 CLASS(UNIXMAP) ID(your-userid) DELETE
```

If your environment has the SETROPTS NOADDCREATOR option in effect, the second PERMIT command is not necessary because RDEFINE does not put the profile creator on the access list.

If you are unable to determine the user name or group name from a RACF message, look at the output from the database unload utility (IRRDBU00) to find the user ID or group associated with a given UID or GID. The mapping profiles should then be added, changed, or deleted as appropriate to be consistent.

Using UNIXPRIV Class Profiles to Manage z/OS UNIX Privileges

You can define profiles in the UNIXPRIV class to grant RACF authorization for certain z/OS UNIX privileges. These privileges are automatically granted to all users with z/OS UNIX superuser authority. By defining profiles in the UNIXPRIV class, you may specifically grant certain superuser privileges with a high degree of granularity to users who do not have superuser authority. This allows you to minimize the number of assignments of superuser authority at your installation and reduces your security risk.

Resource names in the UNIXPRIV class are associated with z/OS UNIX privileges. You must define profiles in the UNIXPRIV class protecting these resources in order to use RACF authorization to grant z/OS UNIX privileges. The UNIXPRIV class must be active and SETROPTS RACLIST must be in effect for the UNIXPRIV class. Global access checking is not used for authorization checking to UNIXPRIV resources.

See *z/OS UNIX System Services Planning* for a list of the resource names available in the UNIXPRIV class, the z/OS UNIX privilege associated with each resource, and the level of access required to grant the privilege.

Example of Authorizing Superuser Privileges

The following examples apply to superuser privileges, except the privilege associated with the CHOWN.UNRESTRICTED resource (see “Using the

CHOWN.UNRESTRICTED Profile"). For example, these are the steps to authorize selected users to transfer ownership of any file.

1. Define a profile in the UNIXPRIV class to protect the resource called SUPERUSER.FILESYS.CHOWN. For example:

   ```
   RDEFINE UNIXPRIV SUPERUSER.FILESYS.CHOWN UACC(NONE)
   ```

   Note: Generic profile names are allowed for resources in the UNIXPRIV class, with the exception of the CHOWN.UNRESTRICTED resource. For example, if you wish to authorize all file-system privileges, you can define a profile called SUPERUSER.FILESYS.**.

2. Authorize selected users or groups as appropriate:

   ```
   PERMIT SUPERUSER.FILESYS.CHOWN CLASS(UNIXPRIV)
          ID(appropriate-groups-and-users) ACCESS(READ)
   ```

3. Activate the UNIXPRIV class, if it is not currently active at your installation:

   ```
   SETROPTS CLASSACT(UNIXPRIV)
   ```

   Note: If you do not activate the UNIXPRIV class and activate SETROPTS RACLIST processing for the UNIXPRIV class, only superusers will be allowed to transfer ownership of any file.

4. You *must* activate SETROPTS RACLIST processing for the UNIXPRIV class, if it is not already active. For a complete description of this function, see "SETROPTS RACLIST Processing" on page 131.

   ```
   SETROPTS RACLIST(UNIXPRIV)
   ```

   Note: If SETROPTS RACLIST processing is already in effect for the UNIXPRIV class, you must refresh SETROPTS RACLIST processing in order for new or changed profiles in the UNIXPRIV class to take effect.

   ```
   SETROPTS RACLIST(UNIXPRIV) REFRESH
   ```

Allowing z/OS UNIX Users to Change File Ownerships

On z/OS UNIX systems, RACF enforces the rules for the POSIX constant called _POSIX_CHOWN_RESTRICTED. This means that, by default, only superusers can change the ownership of any file to any UID or GID on the system, and that general users can only change the ownership of files that they own, and only to one of their own associated GIDs. By defining a profile called CHOWN.UNRESTRICTED, you can indicate that _POSIX_CHOWN_RESTRICTED is not effect. This allows all z/OS UNIX users to transfer ownership of files they own to any UID or GID on the system.

You can define an additional profile in the UNIXPRIV class protecting a resource called SUPERUSER.FILESYS.CHOWN to authorize selected z/OS UNIX users to transfer ownership of any file to any UID or GID. See "Example of Authorizing Superuser Privileges" on page 511 for an example of authorizing users using the SUPERUSER.FILESYS.CHOWN resource.

Using the CHOWN.UNRESTRICTED Profile

To allow all z/OS UNIX users to transfer ownership of files they own to any UID or GID on the system, create a discrete profile in the UNIXPRIV class called CHOWN.UNRESTRICTED. If this profile is defined on your system, _POSIX_CHOWN_RESTRICTED will not be in effect, and all z/OS UNIX users will be allowed to issue the chown command to transfer ownership of files that they own.

The profile that you define in the UNIXPRIV class called CHOWN.UNRESTRICTED must be a discrete profile. Any matching generic profiles will be ignored. No access list is needed for this profile. RACF checks only for the existence of this profile. Any access list will be ignored.

To allow z/OS UNIX users to transfer ownership for files they own, take the following steps:

1. Define the discrete profile in the UNIXPRIV class called CHOWN.UNRESTRICTED:

   ```
   RDEFINE UNIXPRIV CHOWN.UNRESTRICTED
   ```

2. Activate the UNIXPRIV class, if it is not currently active at your installation:

   ```
   SETROPTS CLASSACT(UNIXPRIV)
   ```

   Note: If you do not activate the UNIXPRIV class and activate SETROPTS RACLIST processing for the UNIXPRIV class, only superusers will be allowed to transfer ownership of files to others.

3. You *must* activate SETROPTS RACLIST processing for the UNIXPRIV class, if it is not already active. For a complete description of this function, see "SETROPTS RACLIST Processing" on page 131.

   ```
   SETROPTS RACLIST(UNIXPRIV)
   ```

   Note: If SETROPTS RACLIST processing is already in effect for the UNIXPRIV class, you must refresh SETROPTS RACLIST processing in order for the CHOWN.UNRESTRICTED profile to take effect.

   ```
   SETROPTS RACLIST(UNIXPRIV) REFRESH
   ```

Protecting HFS Data

z/OS UNIX has a hierarchical file system (HFS) that can be thought of as a tree of directories, starting at a root and proceeding from there. Each directory can contain files or other directories. Also, the security rules for the files and directories are stored within the file system itself in the file security packet (FSP).

HFS files and directories are protected by permission bit information that is kept within the FSP. These permission bits allow specification of read authority, write authority, or search authority for a directory. They also allow specification of read, write, or execute for a file. There are three sets of bits so that separate authorities can be specified for the owner of the file or directory, the owning group, and everyone else (like RACF's universal access authority, or UACC). The owner is represented by a UID. The owning group is represented by an GID. Access checking compares the user's UID and GID to the ones stored in the FSP.

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. RACF does not provide commands to maintain the FSP, but it does provide SAF services that do the FSP maintenance. z/OS UNIX provides commands that invoke these SAF services.

See *z/OS UNIX System Services Planning* for information on setting the file permission bits and for information on using RACF authorization to grant privileges for HFS files and directories.

See *z/OS UNIX System Services Command Reference* for more information about these z/OS UNIX commands.

See *z/OS UNIX System Services Programming: Assembler Callable Services Reference* for information on callable services you can use to maintain the FSP.

z/OS UNIX Application Considerations

z/OS UNIX supports two fundamental types of application servers:
- Multithreaded application servers
- Single-threaded application servers

A *multithreaded* application has multiple sequential flows of control. In this family of applications, the application server can process more than one unit of work at a time.

A *single-threaded* application has one sequential flow of control. In this family of applications, the application server processes one unit of work at a time.

z/OS UNIX provides the `pthread_security_np` callable service and support through the C run-time library that enables *unauthorized* multithreaded applications to create and delete a RACF security environment in a fashion that is mediated and controlled by the kernel and RACF.

Note: The term *unauthorized* refers to applications that are not APF-authorized and do not run in supervisor state or in a system storage protection key.

The `pthread_security_np` callable service enables multithreaded applications to customize the security environment of a thread, allowing it to execute under a different RACF identity than the server. You need to authorize the server to use the `pthread_security_np` service.

For More Information

Administrative considerations of the z/OS UNIX `pthread_security_np` callable service are discussed in *z/OS UNIX System Services Planning*.

Additional information regarding the `pthread_security_np` service is discussed in *z/OS UNIX System Services Programming: Assembler Callable Services Reference*. The C language support for this service is discussed in *z/OS C/C++ Run-Time Library Reference*.

Threads and Security

An application that uses the `pthread_security_np` service can customize the RACF identity of a thread. For example, a DCE application server on z/OS may accept requests through DCE remote procedure call (RPC). This server initiates a thread that processes the client's request. If the server customizes the thread initiated for the client with the client's RACF identity, any resource access decisions to RACF protected resources are made using the client's RACF identity and authorizations.

Depending on the trust you place in an application, you have the option of enforcing whether to use both the application server's RACF identity *and* the RACF identity of the client in resource access control decisions.

You can choose one of the following:
- Only the RACF user ID of the client is used in local resource access control decisions made by RACF.

- Both the RACF user ID of the server and the RACF user ID of the client are used in local resource access control decisions.

The use of the `pthread_security_np` service is in part protected by the RACF FACILITY class profile BPX.SERVER.

- If the RACF user ID that is associated with an application server is permitted with UPDATE access to this profile, the application server is allowed to establish a thread-level (task-level) security environment for clients connecting to the server. With UPDATE authority to BPX.SERVER in the RACF FACILITY class, the server can act as a *surrogate* of the client. This means that the identity of the thread associated with the request from the server's client executes with the RACF user ID of the server's client.

  The RACF identity of the client determines the type of access allowed to system resources (such as data sets) and z/OS UNIX resources (such as HFS files), which are accessed by the client's thread in the server.
- READ access allows the server to establish a thread-level security environment for the clients it services. However, the user ID of the server and the user ID of the client must be authorized to the resources the server accesses. A thread level security environment in which both the client's and server's identities are used in the access control decision, but a password was not supplied by the client, is called an *unauthenticated client* security environment.

  Depending on the design and implementation of the client/server application, a client might need to supply an authenticator to the server.

  For example, the client might be prompted to supply a password or a password substitute, such as a RACF PassTicket, to the server to prove its identity. If a RACF password or PassTicket is specified as a option on the `pthread_security_np` service, and the password or PassTicket is valid for the client user ID, *only* the RACF user ID of the client is used in rendering access control decisions. This task level security environment created by an application server is called an *authenticated client* security environment. Because the client has trusted the application server sufficiently to supply a RACF password or PassTicket to the server, the server is granted the capability of acting as a surrogate for that client.

  This capability enables you to determine:

  – On behalf of which user IDs the server can act
  – What resources the server can access when acting on behalf of one of its clients

Potentially, for additional security checking, two audit records can be produced to audit:

- The client accessing the resource
- The server accessing the resource on behalf of the client

If you choose to implement this additional security checking, you might need to authorize the identity associated with the application server to the resource profiles that protect the resources accessed by the server on behalf of its clients.

See *z/OS UNIX System Services Planning* for a complete description of the administrative planning steps and requirements for using the `pthread_security_service`.

Application Services and Security

Through z/OS UNIX, the C run-time library, and RACF, three services are available that enable application servers on z/OS to:

- Map a DCE identity to a RACF user ID; or map a RACF user ID to a DCE identity.

  See “Chapter 12. RACF and DCE” on page 417 for more information.
- Map an application identity, such as a Lotus Notes for z/OS short name or an Novell Directory Services for OS/390 user name, to a RACF user ID; or map a RACF user ID to an application identity.

  See “RACF Support for NDS and Lotus Notes for z/OS” on page 295 for more information.
- Invoke RACF authorization services

The `convert_id_np(BPX1CID)` callable service is the z/OS UNIX service that converts a DCE principal’s UUID pair (cell UUID and principal UUID) to the RACF user ID that has been cross linked with the UUID pair. This service also accepts a RACF user ID and returns the corresponding DCE UUIDs that have been cross-linked with the RACF user ID. This z/OS UNIX service is also supported through the C run-time library via the `__convert_id_np()` function call.

These services use the SAF callable service, `R_dceruid` (IRRSUD00), which accesses the appropriate profile information stored in the RACF database, to perform the identity conversion. The use of these identity mapping functions is RACF-protected. The `R_dceruid` callable service performs an authorization request to determine if the user ID associated with the application server is authorized to use the identity conversion service. Controlling the use of these conversion services is discussed in “Controlling the R_dceruid Callable Service” on page 517.

For more information about the `convert_id_np(BPX1CID)` callable service, see *z/OS UNIX System Services Programming: Assembler Callable Services Reference*. The C language support for the `__convert_id_np()` is discussed in *z/OS C/C++ Run-Time Library Reference*.

Application Authorization Service

A DCE application server on z/OS can use DCE security services for access control to resources that are owned by the application server. As an alternative, the application developer may wish to use RACF for access control for the set of resources that are managed by the application server.

Through z/OS UNIX, the `auth_check_resource_np(BPX1ACK)` callable service enables application servers to invoke RACF authorization services. This service is also supported by the C run-time library through the `__check_resource_auth_np()` function call.

The service allows z/OS UNIX application servers, such as DCE application servers, to perform authorization checking for resources that are defined to RACF general resource classes. For more information on the `auth_check_resource_np` callable service, see *z/OS UNIX System Services Programming: Assembler Callable Services Reference*.

Restrictions of RACF Client ACEE Support

As the security administrator, you need to be aware of restrictions of the RACF client ACEE support, in which both the application server’s RACF identity and the client’s RACF identity are used in resolving access decisions.

- RACROUTE REQUEST=FASTAUTH processing does not check both the server and client RACF identities automatically.

  Unauthorized application servers cannot use the RACROUTE REQUEST=LIST instruction to build in-storage profiles for RACF defined resources. Profiles must reside in storage before RACROUTE REQUEST=FASTAUTH can verify a user's access to a resource.
- The client/server relationship is not propagated from the application server.

If you have implemented access control to resources that use both the server's RACF identity and the client's RACF identity in an access control decision, application servers that you do not trust should be treated as *end points*. These servers should not be allowed to submit batch jobs or use the services of other servers that run exclusively under the identity of the client. You must ensure that applications servers that do not meet this criteria are not authorized to the profile BPX.SERVER in the RACF FACILITY class.

Controlling the R_dceruid Callable Service

You need to define the IRR.RDCERUID profile in the FACILITY class to control the use of the SAF `R_dceruid` callable service. This maps the DCE UUID to the RACF user ID. Applications that use this service must have READ access or higher to the resource IRR.RDCERUID in the RACF FACILITY class.

Attention

You should review product or application installation instructions for products that use:

- The SAF `R_dceruid` callable service
- The z/OS UNIX `convert_id_np` callable service
- The C library function `__convert_id_np()` function call

Chapter 19. RACF and Digital Certificates

This chapter provides information on using RACF to manage digital certificates.

In a client-server network environment, entities identify themselves using *digital certificates.* Certificate authorities (such as VeriSign, Inc.) issue digital certificates that contain information about clients. The combination of a serial number and the name of the certificate authority (or *issuer's distinguished name*) uniquely identifies a client's digital certificate.

Certificate authorities are trusted organizations that verify information about client users and then issue digital certificates that may be accepted by applications as authentication of client identities when used in a secure handshaking protocol such as secure sockets layer (SSL). Trusting a certificate issued by certificate authority is analogous to accepting a passport issued by a national passport agency as proof of identity. We trust that the agency has taken proper measures to verify the identity of the bearer of the passport. In a similar manner, application servers may accept certificate-authority certificates.

In the client-server environment, for example, WebSphere Application Server authenticates a client using the client's digital certificate and the secure sockets layer (SSL) protocol. After a user supplies the digital certificate, WebSphere Application Server checks that the certificate is valid. If the certificate is registered and is not marked as untrusted, the user does not need to be reauthenticated by entering a RACF user ID and password.

Using RACF to Manage Digital Certificates

You can use RACF to create, register, store, and administer digital certificates and their associated private keys, and build certificate requests that can be sent to a certificate authority for signing. You can also use RACF to manage key rings of stored digital certificates. Digital certificates and key rings are managed in RACF primarily by using the RACDCERT command or by using an application that invokes the `R_datalib` callable service (`IRRSDL00`) or the `initACEE` callable service (`IRRSIA00`).

The `R_datalib` callable service provides an application programming interface to the CDSA (Common Data Security Architecture) data library functions, and is used by secure sockets layer (SSL) and System SSL to establish secure sessions between servers. The `initACEE` callable service can be used to manage digital certificates for RACF-authenticated users.

RACF allows you to manage three types of digital certificates:

User certificate
: A certificate that is associated with a RACF user ID and is used to authenticate the user's identity

Certificate-authority certificate
: A certificate that is associated with a certificate authority and is used to verify signatures in other certificates

Site certificate
A certificate that is associated with a server, or network entity other than a user or certificate authority.

You can use RACF to administer certificates in two ways:

- You can register and store an individual certificate, associating it with a specific user ID using the RACDCERT ADD command.

  Registered certificates are stored in general resource profiles in the DIGTCERT class. You can also connect certificates to key rings. Key rings are stored in general resource profiles in the DIGTRING class. For details, see “Using the RACDCERT Command to Administer Certificates”.
- You can map several certificates, without storing them, to one or more RACF user IDs that can be shared by several users, using the RACDCERT MAP command. This method is called *certificate name filtering.*

  Certificate name filters are stored in general resource profiles in the DIGTNMAP class.

Using the RACDCERT Command to Administer Certificates

The RACDCERT command allows a RACF-defined user ID to use a digital certificate as identification. The certificate must be a standard X.509 certificate (such as issued by VeriSign, Inc.) contained in an MVS data set. RACDCERT is used to store and maintain digital certificates in RACF, and should be used for all maintenance of the DIGTCERT class profiles and related USER profile fields.

The RACDCERT command can be used to do the following:

- Add, list, modify or delete a certificate definition.
- List a certificate contained in a data set.
- Add or remove a certificate from a key ring.
- Create, delete or list a key ring.
- Generate a public/private key pair and certificate.
- Create a certificate request.
- Write a certificate to a data set.
- Add, list, modify or delete a certificate name filter.

The RACDCERT command is your primary administrative tool for managing digital certificates using RACF. Authority to use the RACDCERT command is controlled through resources in the FACILITY class. The RACDCERT command is used to manage resources in the following classes:

- DIGTCERT

  Profiles in the DIGTCERT class contain information about digital certificates, as well as the certificate itself.
- DIGTRING

  Profiles in the DIGTRING class contain information about key rings and the certificates that are part of each key ring. Key rings are named collections of the personal, site and certificate-authority certificates associated with a specific user.
- DIGTNMAP

  Profiles in the DIGTNMAP class contain information about certificate name filters.

Note: Profiles in the DIGTCERT, DIGTRING, and DIGTNMAP classes are maintained automatically through RACDCERT command processing. You

cannot administer profiles in these classes using the RDEFINE, RALTER, and RDELETE commands. These commands do not operate with profiles in the DIGTCERT, DIGTRING, and DIGTNMAP classes. Since these profiles contain lowercase characters, the SEARCH FILTER and RLIST commands are not intended for use and will deliver unpredictable results.

- USER

  Profiles in the USER class contain information about digital certificates that are associated with the user.

Attention

Use caution when executing the DELUSER command from downlevel systems if your installation shares the RACF database. A DELUSER command executed from a downlevel system will not manage profiles in the DIGTCERT, DIGTRING, DIGTNMAP and USER class correctly. You may inadvertently create inconsistencies in your RACF database, and may leave residual DIGTCERT, DIGTRING, or DIGTNMAP profiles. See “Using the RACF Remove ID Utility (IRRRID00)” on page 342 for information about locating and removing residual profiles.

See *z/OS SecureWay Security Server RACF Command Language Reference* for more information about the RACDCERT command.

See “RRSF Considerations for Digital Certificates” on page 382 for information about propagating updates made by the RACDCERT command to other nodes in an RRSF network.

Controlling the Use of the RACDCERT Command

Authority to the IRR.DIGTCERT.*function* resource in the FACILITY class allows a user to issue the RACDCERT command. To issue the RACDCERT command, users must have one of the following authorities:

- The SPECIAL attribute
- Sufficient authority to resource IRR.DIGTCERT.*function* in the FACILITY class.
  - READ access to IRR.DIGTCERT.*function* to issue the RACDCERT command for themselves.
  - UPDATE access to IRR.DIGTCERT.*function* to issue the RACDCERT command for others.
  - CONTROL access to IRR.DIGTCERT.*function* to issue the RACDCERT command for SITE and CERTAUTH certificates. (This authority also has other uses.)

For detailed information about the RACDCERT command and the authority required to execute each operand, see *z/OS SecureWay Security Server RACF Command Language Reference*.

Note that users who have insufficient authority to the IRR.DIGTCERT.LIST resource can use the RACDCERT CHECKCERT command to display some digital-certificate information if they have READ authority to the data set containing the certificate. The output they receive reflects only the certificate information contained in the data set. Because they lack sufficient authority to the IRR.DIGTCERT.LIST resource, they will not receive certificate information contained in the RACF database, such

as the TRUST status, the LABEL, or the RACF user ID associated with the certificate. For an example of this output, see “Examples of Checking Digital Certificate Information” on page 528.

Examples of Controlling the Use of the RACDCERT Command

Effective use of RACDCERT requires that its privileges be carefully controlled. However, end-users and application administrators should be allowed some flexibility in defining their security characteristics. These guidelines might prove useful.

- The ability to add certificate authorities and site certificates should be allowed to only a small set of trusted people.
- End users should be permitted to add, delete, and modify the contents of their own key rings and add, delete, and alter their own certificates.
- Help desk personnel should be allowed the ability to list certificates and rings.

Assume that your system administrators, who are the only ones who are allowed to add, alter, or delete certificate-authority certificates or site certificates, are in the group WEBADMIN. Furthermore, assume that your help desk personnel are in the group HELPDESK. The commands in Figure 55 on page 524 show one method of controlling access to RACDCERT functions. Note that similar authorizations may be defined to allow system administrators and help desk personnel to manage certificate name filters.

```
RDEFINE  FACILITY  IRR.DIGTCERT.ADD        UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.ADDRING    UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.DELRING    UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.LISTRING   UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.CONNECT    UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.REMOVE     UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.LIST       UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.ALTER      UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.DELETE     UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.GENCERT    UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.GENREQ     UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.EXPORT     UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.MAP        UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.ALTMAP     UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.DELMAP     UACC(NONE)
RDEFINE  FACILITY  IRR.DIGTCERT.LISTMAP    UACC(NONE)

PERMIT   IRR.DIGTCERT.ADDRING  CLASS(FACILITY) ID(WEBADMIN) ACCESS(CONTROL)
PERMIT   IRR.DIGTCERT.DELRING  CLASS(FACILITY) ID(WEBADMIN) ACCESS(CONTROL)
PERMIT   IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(WEBADMIN) ACCESS(CONTROL)
PERMIT   IRR.DIGTCERT.ADD      CLASS(FACILITY) ID(WEBADMIN) ACCESS(CONTROL)
PERMIT   IRR.DIGTCERT.ALTER    CLASS(FACILITY) ID(WEBADMIN) ACCESS(CONTROL)
PERMIT   IRR.DIGTCERT.DELETE   CLASS(FACILITY) ID(WEBADMIN) ACCESS(CONTROL)
PERMIT   IRR.DIGTCERT.LIST     CLASS(FACILITY) ID(WEBADMIN) ACCESS(CONTROL)
PERMIT   IRR.DIGTCERT.GENCERT  CLASS(FACILITY) ID(WEBADMIN) ACCESS(CONTROL)
PERMIT   IRR.DIGTCERT.GENREQ   CLASS(FACILITY) ID(WEBADMIN) ACCESS(CONTROL)
PERMIT   IRR.DIGTCERT.EXPORT   CLASS(FACILITY) ID(WEBADMIN) ACCESS(CONTROL)

PERMIT   IRR.DIGTCERT.ADD      CLASS(FACILITY) ID(*)
PERMIT   IRR.DIGTCERT.ALTER    CLASS(FACILITY) ID(*)
PERMIT   IRR.DIGTCERT.DELETE   CLASS(FACILITY) ID(*)
PERMIT   IRR.DIGTCERT.LIST     CLASS(FACILITY) ID(*)
PERMIT   IRR.DIGTCERT.ADDRING  CLASS(FACILITY) ID(*)
PERMIT   IRR.DIGTCERT.DELRING  CLASS(FACILITY) ID(*)
PERMIT   IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(*)
PERMIT   IRR.DIGTCERT.CONNECT  CLASS(FACILITY) ID(*)
PERMIT   IRR.DIGTCERT.REMOVE   CLASS(FACILITY) ID(*)

PERMIT   IRR.DIGTCERT.LIST     CLASS(FACILITY) ID(HELPDESK) ACCESS(CONTROL)
PERMIT   IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(HELPDESK) ACCESS(CONTROL)
```

Figure 55. Controlling Access to RACDCERT Functions

Examples of Adding Digital Certificate Information

1. User RACFADM with SPECIAL authority requests the addition of a digital certificate for user NET2. RACFADM has placed the digital certificate in data set 'RACFADM.NET2.CERT' and wants the status of the certificate to be trusted. RACFADM issues the following RACDCERT command:

   ```
   RACDCERT ID(NET2) ADD('RACFADM.NET2.CERT') TRUST
   ```

2. User RACFADM with SPECIAL authority requests the addition of a digital certificate for user NETB0Y. RACFADM has placed the digital certificate in data set 'RACFADM.NETB0Y.CERT' and wants the status of the certificate to be trusted. In addition, RACFADM wants to associate the saved certificate for user NETB0Y with a label called ″Savings Account″. RACFADM issues the following RACDCERT command:

   ```
   RACDCERT ID(NETB0Y) ADD('RACFADM.NETB0Y.CERT') TRUST
            WITHLABEL('Savings Account')
   ```

Examples of Listing Digital Certificate Information

1. User RACFADM with SPECIAL authority requests the listing of user ID GEORGEM's digital certificate information by issuing the RACDCERT command with the LIST operand. User ID GEORGEM has three certificates, one of which is not associated with any key rings. Figure 56 on page 526 shows the output of the following command:

   ```
   RACDCERT ID(GEORGEM) LIST
   ```

```
Digital certificate information for user GEORGEM:

   Label: New Cert Type - Ser # 00
   Status: TRUST
   Start Date: 1996/04/18 03:01:13
   End Date:   1998/02/13 03:01:13
   Serial Number:
       >00<
   Issuer's Name:
       >OU=Internet Demo CA.O=TheCert Software Inc.<
   Subject's Name:
      >OU=Internet Demo CA.O=TheCert Software Inc.<
   Private Key Type: ICSF
   Private Key Size: 1024
   Ring Associations:
   Ring Owner: GEORGEM
   Ring:
      >GEORGEMsNewRing01<
   Ring Owner: GEORGEM
   Ring:
      >GEORGEMsRing<

   Label: New Type Cert - VsignC1
   Status: TRUST
   Start Date: 1998/04/22 23:23:26
   End Date:   2001/01/15 23:23:26
   Serial Number:
      >3511A552906FE7D029A44019D411FC3E<
   Issuer's Name:
   >OU=Class 1 Public Primary Certification Authority.O=VeriSign,  Inc..C=<
   >US<
   Subject's Name:
   >OU=VeriSign Class 1 CA - Individual Subscriber.O=VeriSign, Inc..L=Int<
   >ernet<
   Private Key Type: Non-ICSF
   Private Key Size: 512
   Ring Associations:
   Ring Owner: GEORGEM
   Ring:
       >GEORGEMsNewRing01<

   Label: New Type Cert - VsignC2
   Status: NOTRUST
   Start Date: 1998/03/19 15:39:52
   End Date:   1999/03/19 15:39:52
   Serial Number:
       >50D35294912F79D315E32B31AC8548F0<
   Issuer's Name:
       >OU=Class 2 Public Primary Certification Authority.O=VeriSign, Inc..C=<
       >US<
   Subject's Name:
       >OU=VeriSign Class 2 CA - Individual Subscriber.O=VeriSign, Inc..L=Int<
       >ernet<
       Private Key Type: None
       Ring Associations:
       *** No rings associated ***
```

Figure 56. Output from the RACDCERT LIST Command

2. User RACFADM with SPECIAL authority requests the listing of user ID GEORGEM's key rings by issuing the RACDCERT command with the LISTRING operand. User ID GEORGEM has three key rings with certificates and one key ring which has no certificates. Figure 57 on page 527 shows the output of the following command:

```
RACDCERT ID(GEORGEM) LISTRING
```

```
Digital ring information for user GEORGEM:

   Ring:
        >GEORGEMsNewRing01<
   Certificate Label Name              Cert Owner     USAGE      DEFAULT
   --------------------------------    ------------   --------   -------
   New Cert Type - Ser # 00            ID(GEORGEM)    PERSONAL     YES
   New Type Cert - VsignC1             ID(GEORGEM)    CERTAUTH     NO
   New Type Cert - VsignC2             ID(GEORGEM)    SITE         NO
   65                                  ID(JOHNP)      PERSONAL     NO

   Ring:
        >GEORGEMsRing<
   Certificate Label Name              Cert Owner     USAGE      DEFAULT
   --------------------------------    ------------   --------   -------
   GEORGEM's Cert # 48                 ID(GEORGEM)    PERSONAL     NO
   GEORGEM's Cert # 84                 ID(GEORGEM)    PERSONAL     NO
   New Cert Type - Ser # 00            ID(GEORGEM)    PERSONAL     YES

   Ring:
        >GEORGEMsRing#2<
   Certificate Label Name              Cert Owner     USAGE      DEFAULT
   --------------------------------    ------------   --------   -------
   GEORGEM's Cert # 84                 ID(GEORGEM)    PERSONAL     NO
   GEORGEM's Cert # 48                 ID(GEORGEM)    PERSONAL     NO

   Ring:
        >GEORGEMsRing#3<
   *** No certificates connected ***
```

Figure 57. Output from the RACDCERT LISTRING Command

3. User NETB0Y requests the listing of his Savings Account digital certificate to ensure it has been defined, and that it is marked trusted. He has READ authority to the FACILITY class resource IRR.DIGTCERT.LIST. He issues the RACDCERT command with the LIST operand, specifying the label to identify his certificate:

```
RACDCERT LIST(LABEL('Savings Account'))

  Digital certificate information for user NETB0Y:

   Label: Savings Account
   Status: TRUST
   Serial Number:
        >5D666C20207A6638727A413872D8413B<
   Issuer's Name:
        >OU=BobsBank Savers.O=BobsBank.L=Internet<
   Subject's Name:
        >CN=S.S.Smith.OU=Digital ID Class 1 - NetScape.OU=BobsBank Class 1 - S<
        >avingsAcct.O=BobsBank.L=Internet<
```

Examples of Checking Digital Certificate Information

1. User NETADMN has a digital certificate in a data set, and is uncertain who it belongs to, and whether or not it has been defined. The digital certificate is in data set 'NETADMN.SOMEONZ.CERT'. NETADMN has UPDATE authority to the FACILITY class resource IRR.DIGTCERT.LIST. He issues the following RACDCERT, and the output he receives indicates that it has already been defined for user GTM:

```
RACDCERT CHECKCERT('NETADMN.SOMEONZ.CERT')

  Digital certificate information for user GTM:

   Label: LABEL00000001
   Status: NOTRUST
   Start Date: 1998/06/05 14:58:37
   End Date:   2000/06/04 14:58:37
   Serial Number:
        >84<
   Issuer's Name:
        >CN=BobsBank Class 2<
   Subject's Name:
        >loanOf@BobsBank.com.CN=G.T.Miles.T=President.OU=Loans.O=BobsBank,INC<
        >..SP=NY.L=Internet.C=USA<
   Private Key Type: ICSF
   Private Key Size: 1024
```

2. User USERA finds a digital certificate and is uncertain who it belongs to, and whether or not it has been defined to RACF. The digital certificate is contained in data set 'NETADMN.SOMEONZ.CERT' and is associated with user GTM. USERA has READ authority to the data set 'NETADMN.SOMEONZ.CERT'. He issues the following RACDCERT command. The output he receives reflects only the certificate information contained in the data set, and does not include certificate information contained in the RACF database. Note that the listing contains the same level of information that NETADMN receives in example 3.

```
RACDCERT CHECKCERT('NETADMN.SOMEONZ.CERT')

   Start Date: 1998/06/05 14:58:37
   End Date:   2000/06/04 14:58:37
   Serial Number:
        >84<
   Issuer's Name:
        >CN=BobsBank Class 2<
   Subject's Name:
        >loanOf@BobsBank.com.CN=G.T.Miles.T=President.OU=Loans.O=BobsBank,INC<
        >..SP=NY.L=Internet.C=USA<
```

3. User NETADMN has a digital certificate in a data set, and is uncertain who it belongs to, and whether or not it has been defined. The digital certificate is in data set 'NETADMN.SOMELSZ.CERT'. NETADMN has CONTROL authority to the FACILITY class resource IRR.DIGTCERT.LIST. He issues the following RACDCERT, and the output he receives indicates that the certificate is not associated with a user ID.

```
RACDCERT CHECKCERT('NETADMN.SOMELSZ.CERT')

   Start Date: 1998/03/18 14:58:37
   End Date:   2000/03/17 14:58:37
   Serial Number:
        >79<
   Issuer's Name:
        >CN=BobsBank Class 2<
   Subject's Name:
        >brchMGR@BobsBank.com.CN=J. Miles.T=Manager.OU=Branch2.O=BobsBank,INC<
        >..SP=NY.L=Internet.C=USA<
```

Examples of Altering Digital Certificate Information

Using the TRUST Option

User CERTCTL requests that the status of user ID NET1's certificate from VeriSign be changed from not trusted to trusted. Because NET1 has more than one certificate defined, the serial number must be specified. The issuer's distinguished name may also be specified, but is not required because each of the two serial numbers for user NET1 is unique. User CERTCTL has been given UPDATE access to resource IRR.DIGTCERT.ALTER in the FACILITY class by RACFADM. Note that when the distinguished name is specified, the case used must match the actual digital certificate information:

```
RACDCERT ID(NET1) ALTER(SERIALNUMBER(41D87A3B05DE6FBD466C2069661E3872)
                        ISSUERSDN('OU=VeriSign Class 2 CA - Individual
                                  Subscriber.O=VeriSign, Inc..L=Internet')) TRUST
```

Using the NOTRUST Option

User CERTCTL requests that the status of user ID NET2's certificate from VeriSign be changed from trusted to not trusted. Because user NET2 has only one certificate defined, neither the serial number nor the issuer's distinguished name needs to be specified. User CERTCTL has been given UPDATE access to resource IRR.DIGTCERT.ALTER in the FACILITY class by RACFADM.

```
RACDCERT ID(NET2) ALTER NOTRUST
```

RACF and Key Rings

A key ring is a collection of certificates that identify a networking trust relationship (also called a trust policy). In a client-server network environment, entities identify themselves using digital certificates. Server applications on z/OS and OS/390 that wish to establish network connections to other entities can use RACF key rings and other related services to determine the trustworthiness of the client or peer entity.

The usage assigned to a certificate when it is connected to a key ring indicates its intended purpose. Personal certificates are to be used by the local server application to identify itself. Certificate-authority certificates are to be used to verify the peer entity's certificate. Peers with certificates issued by certificate authorities connected to the key ring are considered trusted network entities. Peers possessing certificates that can not be verified because the certificate-authority certificate is not available may also be considered trusted if their personal certificates are connected to the key ring as a trusted site certificate.

Note: Use caution when connecting a peer's certificate to a key ring as a trusted site certificate. The normal certificate verification tests performed by the server on the peer's certificate are bypassed in this case. Hence, even expired certificates are considered trusted.

Key rings are associated with specific RACF user IDs. A RACF user ID can have more than one key ring. Key rings are managed using the RACDCERT command, and are maintained in the general resource class called DIGTRING.

RACF key rings provide an installation-wide method to share key rings across multiple servers. You can decentralize responsibility to manage key rings by granting access to resources in the FACILITY class. (See "Examples of Controlling the Use of the RACDCERT Command" on page 523.) However, you can retain sole ability to connect certificates to key rings at your installation. This will allow you to implement and maintain a centralized security or trust policy toward certificate authorities. For example, you can establish key rings for servers that contain

certificates from only approved certificate authorities. You can then delegate other key-ring responsibilities to server administrators who will be able remove certificates from their key rings, but not add certificates from unapproved sources.

Key rings are identified by ring names that are 1–237 characters in length. Each key-ring profile in the DIGTRING class contains references to those certificates that are part of that key ring. Profile names are in the form:

userid.ring-name

When you delete a user ID, DELUSER command processing deletes the user's key rings by deleting the associated resources in the DIGTRING class. The certificates referenced in the key ring are not deleted.

DIGTCERT General Resource Class

RACLISTing the DIGTCERT class is strongly recommended to improve performance when using digital certificates to access WebSphere Application Server. If the class is not RACLISTed, digital certificates can still be used, but performance may be impacted. For best performance, you should RACLIST the DIGTCERT class by issuing the SETROPTS RACLIST(DIGTCERT) command. After creating a new digital certificate, you should refresh the DIGTCERT class by issuing the SETROPTS RACLIST(DIGTCERT) REFRESH command. If you do not refresh the RACLISTed DIGTCERT profiles, RACF will still use the new digital certificate. However, performance may be impacted.

Note: RACF will use the RACLISTed digital certificates first, so any RACLISTed digital certificates that have been altered, re-added, or deleted will not reflect those changes until the DIGTCERT class has been refreshed.

Profile names in the DIGTCERT class are in the form:

serial-number.issuer's-distinguished-name

For example:

```
41D87A3B05DE6FBD466C2069661E3872          (serial number)
OU=VeriSign Class1.O=VeriSign.L=Internet  (issuer's distinguished name)
```

Any characters that would not be valid in a profile name, such as a blank, will be replaced with X'4A' (¢). The profile name for this DIGTCERT profile would be:

```
41D87A3B05DE6FBD466C2069661E3872.OU=VeriSign¢Class1.O=VeriSign.L=Internet
```

In this profile, the APPLDATA contains the RACF user ID associated with this digital certificate. The UACC of profiles in this class is the status of the certificate. When a certificate's status is set to TRUST, the `initACEE` callable service allows it to be mapped to the RACF user ID contained in the APPLDATA field.

Certificate Name Filtering

As more and more users access your system from the web, you face an increasing administrative burden to securely manage their digital certificates. *Certificate name filtering* is a method for administering large numbers of user certificates, without storing each certificate in the RACF database. Certificates managed using certificate name filtering:

- Require no individual administration to be registered or to be replaced when they expire.
- Occupy very little space in the RACF database.

- Can be used to allow several users to share the same user ID in a secure manner.
- Can be selectively mapped to different user IDs based on system and application criteria.
- Are logged on use with audit records that include the associated user ID and the certificate's full subject's and issuer's name.

Interpreting the X.500 Directory Information Tree

When you use certificate name filtering, RACF provides the ability to allow several users to share the same user ID on your system based on the *subject's distinguished name* and the *issuer's distinguished name* as contained in X.509 certificates. The subject's distinguished names and issuer's distinguished names for three sample certificates are listed in Table 38 and are shown in the address form used by RACF:

Table 38. Subject's and Issuer's Distinguished Names

Subject's distinguished name	Issuer's distinguished name
CN=Agneta Berglund.OU=Sales.OU=New York.OU=US.O=World Sales Corp	OU=VeriSign Class 1 Individual Subscriber.O=VeriSign, Inc.L=Internet
CN=Hiro Ogura.OU=Admin.OU=New York.OU=US.O=World Sales Corp	OU=VeriSign Class 1 Individual Subscriber.O=VeriSign, Inc.L=Internet
CN=Timo Kokkonen.OU=Sales.OU=Los Angeles.OU=US.O=World Sales Corp	OU=VeriSign Class 1 Individual Subscriber.O=VeriSign, Inc.L=Internet

The distinguished names contained in the certificates shown in Table 38 are represented in the X.500 directory information tree shown in Figure 58.For a list of the components of the subject's X.509 distinguished name, see the syntax of the RACDCERT command in *z/OS SecureWay Security Server RACF Command Language Reference*.

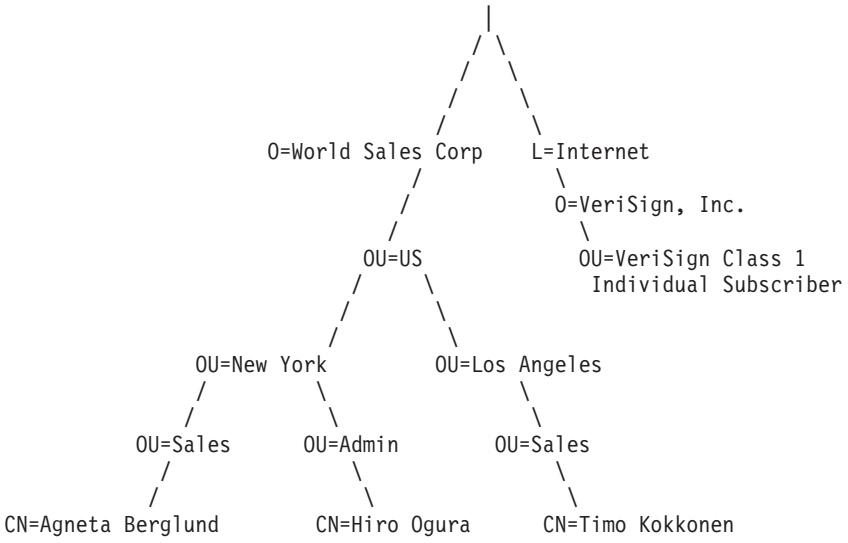


Figure 58. Example of an X.500 Directory Information Tree

Now, let's look at the left branch of the tree in Figure 58 as representing a hierarchical organization, with each level of the tree, or node, representing a

different level within an organization. For example, Agneta works in the Sales department in New York for the US division of the World Sales Corporation. If viewed as a hierarchy of user groups, each level of the tree may represent increased access authority, with each group consisting of the groups below it.

For example, as an employee of World Sales, Agneta may have access to the internal phone numbers of all World Sales employees. As a member of the US division, she may also have access to the US division internal web site, in addition to the phone numbers of all employees. Being in New York may allow her to run sales reports for the New York office, as well as to access the web site and employee phone numbers. Being in the Sales department may allow her to place customer orders, in addition to all other access authorities.

You can associate a user ID with each node in a directory information tree using certificate name filtering. Each user ID may represent a number of users, each of whom has one or more digital certificates. Therefore, you can administer several certificates and the access authorities for several users, through a single user ID. For each node that you associate with a user ID, you create a certificate name filter that contains partial or full distinguished names, depending on where the node falls in the hierarchy.

Creating Certificate Name Filters

You create certificate name filters using the RACDCERT MAP command. *Certificate name filters* are used by RACF (specifically, the `initACEE` callable service) to analyze the subject's and issuer's distinguished names in a given certificate to determine the user ID to associate with it. You can create filters based on the full issuer's distinguished names in order to administer all certificates by a given issuer as a single user ID. You can also create filters based on portions of the subject's distinguished name, and a variety of filters based on certain combinations of partial and full distinguished names. See "Types of Certificate Name Filters" on page 535.

Example:

The RACDCERT MAP command shown in Figure 59 creates a certificate name filter based on the full issuer's distinguished name. This filter associates the user ID `WEBUSER` to any user presenting a certificate issued by VeriSign Class 1, who does not have an individual certificate registered with RACF on your system.

```
RACDCERT ID(WEBUSER) MAP WITHLABEL('INTERNET OTHERS') TRUST
         IDNFILTER('OU=VeriSign Class 1 Individual
                    Subscriber.O=VeriSign, Inc.L=Internet')
SETROPTS RACLIST(DIGTNMAP) REFRESH
```

Figure 59. Sample RACDCERT MAP Command for Creating an Issuer's Name Filter

See *z/OS SecureWay Security Server RACF Command Language Reference* for more information about the RACDCERT MAP command.

Assigning User IDs to Certificate Name Filters

You must define a RACF user ID for each user ID you associate with a certificate name filter. Since these user IDs will be shared, you should consider assigning the PROTECTED and RESTRICTED attributes to each one. For example:

```
ALTUSER WEBUSER NOPASSWORD RESTRICTED
```

The PROTECTED attribute protects the user ID from being used to logon directly to the system and from being revoked through incorrect password attempts. See “Defining Protected User IDs” on page 86.

The RESTRICTED attribute ensures that the user ID will not be used to access protected resources it is not specifically authorized to access. See “Defining Restricted User IDs” on page 87.

Activating Certificate Name Filtering

When you create a certificate name filter, RACDCERT MAP processing automatically creates a mapping profile in the DIGTNMAP class to represent the new filter. The DIGTNMAP class must be active and SETROPTS RACLIST processing must be active for the DIGTNMAP class. Before creating any certificate name filters using the RACDCERT MAP command, you must issue the following command:

```
SETROPTS CLASSACT(DIGTNMAP) RACLIST(DIGTNMAP)
```

Once SETROPTS RACLIST processing is active for the DIGTNMAP class, you must refresh the DIGTNMAP class in order for new or changed certificate name filters to take effect. After creating or changing a certificate name filter, you must issue the following command:

```
SETROPTS RACLIST(DIGTNMAP) REFRESH
```

The DIGTNMAP General Resource Class

RACDCERT MAP processing automatically creates mapping profiles in the DIGTNMAP class for each certificate name filter you create. When you map a certificate name filter to a RACF user ID, both the filter and the user ID are stored in the mapping profile. DIGTNMAP profiles should not be administered using the RDEFINE, RALTER or RDELETE commands. These commands do not operate with the DIGTNMAP class.

Attention

Use caution when executing the RACDCERT MAP command from systems that do not support the DIGTNMAP class, if your installation shares the RACF database with downlevel systems. A RACDCERT command executed from a downlevel system will not manage profiles in the DIGTNMAP class. You may inadvertently create inconsistencies in your RACF database, and may leave residual DIGTNMAP profiles. See “Using the RACF Remove ID Utility (IRRRID00)” on page 342 for more information about locating and removing residual profiles.

The SEARCH FILTER and RLIST commands are not intended for use with profiles in the DIGTNMAP class and will deliver unpredictable results. These profiles can only be displayed using the RACDCERT LISTMAP command: For example:

```
RACDCERT ID(WEBUSER) LISTMAP
```

Based on the output of the RACDCERT LISTMAP command shown in Figure 60 on page 535, there is one certificate name filter associated with the `WEBUSER` user ID.

```
Mapping information for user WEBUSER:
 Label: INTERNET OTHERS
 Status: TRUST
 Issuer's Name Filter:
   >OU=VeriSign Class 1 Individual Subscriber.O=VeriSign, Inc.L=Internet<
 Subject's Name Filter:
   ><
```

Figure 60. Sample Output from the LISTMAP Command for an Issuer's Name Filter

See *z/OS SecureWay Security Server RACF Command Language Reference* for more information about the RACDCERT LISTMAP command.

Types of Certificate Name Filters

There are three basic types of certificate name filters, based on the X.509 distinguished names referenced in the filter definition. They are:

1. Issuer's name filter
2. Subject's name filter
3. Subject's and issuer's name filter

Issuer's Name Filter

An issuer's name filter contains a full or partial issuer's distinguished name. For an example of using the RACDCERT MAP command to create an issuer's name filter, see Figure 59 on page 533. For an example of the output of a RACDCERT LISTMAP command displaying mapping information for an issuer's name filter, see Figure 60.

Subject's Name Filter

A subject's name filter may contain a full or partial subject's distinguished name. Using the directory information shown in Figure 58 on page 532, suppose we want to associate all users in the New York office with the user ID NYUSER, and all users in the New York sales department with user ID NYSALES. We will create two subject's name filters, based on the following significant portions of the subject's distinguished names:

```
OU=Sales.OU=New York.OU=US.O=World Sales Corp

OU=New York.OU=US.O=World Sales Corp
```

Examples: The RACDCERT MAP commands shown in Figure 61 create two subject's name filters based on partial subject's distinguished names.

```
RACDCERT ID(NYSALES) MAP WITHLABEL('NY SALES REPS') TRUST
         SDNFILTER('OU=Sales.OU=New York.OU=US.O=World Sales Corp')
RACDCERT ID(NYUSER)  MAP WITHLABEL('NY OTHERS') TRUST
         SDNFILTER('OU=New York.OU=US.O=World Sales Corp')
SETROPTS RACLIST(DIGTNMAP) REFRESH
```

Figure 61. Sample RACDCERT MAP Commands for Creating Subject's Name Filters

The filter labeled 'NY SALES REPS' contains the portion of the subject's distinguished name that identifies the user as an employee of the Sales department in the New York office of the US division of the World Sales Corporation. Based on this filter, RACF will associate the user ID NYSALES to any user presenting a certificate

containing this significant portion of the subject's distinguished name, who does not have an individual certificate registered with RACF.

The filter labeled 'NY OTHERS' contains the portion of the subject's distinguished name that identifies the user as an employee in the New York office of the US division of the World Sales Corporation. Based on this filter, RACF will associate the user ID NYUSER to any user presenting a certificate containing this significant portion of the subject's distinguished name, who does not have an individual certificate registered with RACF.

Users that present certificates that contain subject's distinguished names that match *both* filters will be associated with the user ID of the *most specific* filter. In this case, the most specific filter is the filter labeled 'NY SALES REPS'. For example, if the users Agneta and Hiro, whose certificate information is shown in Table 38 on page 532, present certificates while these two subject's name filters are in effect, the following will result:

1. Agneta will be associated with the user ID NYSALES, based on the filter labeled 'NY SALES REPS'.
2. Hiro will be associated with the user ID NYUSER, based on the filter labeled 'NY OTHERS'.

   Note: If either Agneta or Hiro had individual certificates registered to RACF, they would have been assigned the user ID specified when the certificates were registered.

Details for Processing Subject's Name Filters: Using the previous example, Hiro presents a certificate that is not registered with RACF. The following represents the sequence of processing that RACF, specifically the initACEE callable service, will complete to process a subject's name filter.

1. The sequence shown in "How RACF Processes Certificate Name Filters" on page 538 is followed, until the full subject's name is used to check for a matching profile in the DIGTNMAP class, to determine if there is an applicable certificate name filter.

   Result: No DIGTNMAP profile is found to match:

   CN=Hiro Ogura.OU=Admin.OU=New York.OU=US.O=World Sales Corp
2. A partial subject's name is formed by removing the most specific node (the CN node) and used to check for a matching profile in the DIGTNMAP class.

   Result: No DIGTNMAP profile is found to match:

   OU=Admin.OU=New York.OU=US.O=World Sales Corp
3. The next partial subject's name is formed by removing the next most specific node (OU=Admin) and used to check for a matching profile in the DIGTNMAP class.

   Result: A DIGTNMAP profile is found to match:

   OU=New York.OU=US.O=World Sales Corp
4. Processing by initACEE continues using the user ID NYUSER for Hiro's certificate.

Subject's and Issuer's Name Filter

A subject's and issuer's name filter contains a combination of a full or partial subject's distinguished name, and the full issuer's distinguished name. These filters can be used when either the subject's name alone or the issuer's name alone does not provide enough information to associate a certificate with a user ID. This

happens when two different certificate authorities issue certificates for the same subject name, or, most commonly, when one certificate authority issues certificates for many different subject names.

A subject's and issuer's name filter can contain the full subject's name, including the CN node, and the full issuer's name. In such a case, you can consider registering the certificate that contains these full names using the RACDCERT ADD command. However, if you register the certificate, RACF will store the certificate as a DIGTCERT profile and you will need to take action when the certificate expires to remove or replace it.

Using the directory information shown in Figure 58 on page 532, suppose we add another filter to our previously defined name filters. This filter will associate all users in the Administration department of the New York office with the user ID NYADMIN. We will create a subject's and issuer's name filter, based on the following significant portion of the subject name, and the full issuer's name:

```
OU=Admin.OU=New York.OU=US.O=World Sales Corp

OU=VeriSign Class 1 Individual Subscriber.O=VeriSign, Inc.L=Internet
```

Example: The RACDCERT MAP command shown in Figure 62 creates a subject's and issuer's name filter based on the partial subject's distinguished name and the full issuer's name.

```
RACDCERT ID(NYADMIN) MAP WITHLABEL('NY ADMIN') TRUST
         SDNFILTER('OU=Admin.OU=New York.OU=US.O=World Sales Corp')
         IDNFILTER('OU=VeriSign Class 1 Individual
                   Subscriber.O=VeriSign, Inc.L=Internet')
SETROPTS RACLIST(DIGTNMAP) REFRESH
```

Figure 62. Sample RACDCERT MAP Command for Creating a Subject's and Issuer's Name Filter

This filter contains the portion of the subject's distinguished name that identifies the user as an employee of the Administration department in the New York office of the US division of the World Sales Corporation, and the full issuer's distinguished name that identifies the issuer as VeriSign Class 1. Based on this filter, RACF will associate the user ID NYADMIN to any user presenting a certificate issued by VeriSign Class 1 containing this significant portion of the subject's distinguished name, who does not have an individual certificate registered with RACF.

Therefore, if the users Timo and Hiro, whose certificate information is shown in Table 38 on page 532, present certificates while all defined name filters are in effect, the following will result:

1. Hiro will be associated with the user ID NYADMIN, based on the filter labeled 'NY ADMIN'.
2. Timo will be associated with the user ID WEBUSER, based on the filter labeled 'INTERNET OTHERS'.

   Note: If either Hiro or Timo had individual certificates registered to RACF, they would have been assigned the user ID specified when the certificates were registered.

Details for Processing Subject's and Issuer's Name Filters: Timo presents a digital certificate that is not registered with RACF. The following represents the sequence of processing that RACF, specifically the `initACEE` callable service, will complete in order to process full and partial subject's names.

1. The sequence shown in "How RACF Processes Certificate Name Filters" is followed, until the full subject's name and issuer's name is used to check for a matching profile in the DIGTNMAP class, to determine if there is an applicable certificate name filter.

   Result: No DIGTNMAP profile is found to match:

   ```
   CN=Timo Kokkonen.OU=Sales.OU=Los Angeles.OU=US.O=World Sales Corp |
   OU=VeriSign Class 1 Individual Subscriber.O=VeriSign, Inc.L=Internet
   ```

2. A partial subject's name is formed by removing the most specific node (the CN node) and used to check for a matching profile in the DIGTNMAP class.

   Result: No DIGTNMAP profile is found to match:

   ```
   OU=Sales.OU=Los Angeles.OU=US.O=World Sales Corp | OU=VeriSign Class 1
   Individual Subscriber.O=VeriSign, Inc.L=Internet
   ```

3. The next partial subject's name is formed by removing the next most specific node (`OU=Sales`) and used to check for a matching profile in the DIGTNMAP class.

   Result: No DIGTNMAP profile is found to match:

   ```
   OU=Los Angeles.OU=US.O=World Sales Corp | OU=VeriSign Class 1 Individual
   Subscriber.O=VeriSign, Inc.L=Internet
   ```

4. The next partial subject's name is formed by removing the next most specific node (`OU=Los Angeles`) and used to check for a matching profile in the DIGTNMAP class.

   Result: No DIGTNMAP profile is found to match:

   ```
   OU=US.O=World Sales Corp | OU=VeriSign Class 1 Individual
   Subscriber.O=VeriSign, Inc.L=Internet
   ```

5. The last partial subject's name is formed by removing the next most specific node (`OU=US`) and used to check for a matching profile in the DIGTNMAP class.

   Result: No DIGTNMAP profile is found to match:

   ```
   O=World Sales Corp | OU=VeriSign Class 1 Individual
   Subscriber.O=VeriSign, Inc.L=Internet
   ```

6. The full issuer's name is then used to check for a matching profile in the DIGTNMAP class.

   Result: A DIGTNMAP profile is found to match:

   ```
   OU=VeriSign Class 1 Individual Subscriber.O=VeriSign, Inc.L=Internet
   ```

7. Processing by `initACEE` continues using the user ID `WEBUSER` for the Timo's certificate.

How RACF Processes Certificate Name Filters

When user presents a digital certificate as identification and the `initACEE` callable service is called to associate the certificate with a user ID, `initACEE` first searches the DIGTCERT class using the certificate's serial number and issuer's distinguished name to see if the certificate was previously registered to RACF. If no match is found in the DIGTCERT class, `initACEE` attempts to locate an appropriate certificate name filter by searching the DIGTNMAP class using a series of full and partial distinguished names until the most specific matching filter is found. If no match is found, and the certificate does not contain a `hostIdMappings` extension (see "Using a hostIdMappings Extension" on page 545), the certificate cannot be used to identity the user to RACF.

The following values are used in sequence to search for a matching certificate name filter:

1. *subject's-full-name.issuer's-full-name*
2. *subject's-partial-name.issuer's-full-name*
3. *subject's-full-name*
4. *subject's-partial-name*
5. *issuer's-full-name*
6. *issuer's-partial-name*

As soon as a matching certificate name filter is found, the user ID associated with the filter is used to identify the user of the certificate. Note that searching is not done for the following values:

subject's-full-name.issuer's-partial-name
subject's-partial-name.issuer's-partial-name

Each step of the search using a partial name may actually involve a series of searches for partial name values based on the full name. Each partial name value in the series is determined by removing the next most specific node in the name. For details on searching for a series of partial name values, see the next example using Timo's certificate.

Using a Stored Certificate as a Model

A digital certificate that is registered to RACF can be used as a model for a certificate name filter, if it is available in a cataloged data set. Using the RACDCERT MAP command with the MAP(*data-set-name*) option, a stored certificate can be used to model the subject's name filter, the issuer's name filter, or both. The subject's distinguished name in the certificate is used beginning with the value specified with the SDNFILTER. The issuer's distinguished name in the certificate is used beginning with the value specified with the IDNFILTER.

For example, let's assume that Ines Soto's certificate is available in data set 'CERTADM.SOTO', and that it contains the following subject's and issuer's names:

```
CN=Ines Soto.OU=Admin.OU=New York.OU=US.O=World Sales Corp

OU=VeriSign Class 1 Individual Subscriber.O=VeriSign, Inc.L=Internet
```

The RACDCERT MAP commands shown in Figure 63 can be used to create certificate name filters using Ines Soto's certificate as a model. Note that only the starting point for each filter needs to be specified to indicate where the filter name should begin.

```
RACDCERT ID(WEBUSER) MAP('CERTADM.SOTO') WITHLABEL('INTERNET OTHERS')
         IDNFILTER('OU=') TRUST
RACDCERT ID(NYUSER)  MAP('CERTADM.SOTO') WITHLABEL('NY OTHERS')
         SDNFILTER('OU=N') TRUST
RACDCERT ID(NYADMIN) MAP('CERTADM.SOTO') WITHLABEL('NY SALES REPS')
         SDNFILTER('OU=')
         IDNFILTER('OU=') TRUST
SETROPTS RACLIST(DIGTNMAP) REFRESH
```

Figure 63. Sample RACDCERT MAP commands using a Model Certificate

The RACDCERT MAP commands in Figure 64 can be used to create the same certificate name filters as those created by the RACDCERT MAP commands in Figure 63 on page 539. Note that the RACDCERT commands in Figure 63 on page 539 using the model certificate are shorter and may minimize typographic errors when defining long filter names.

```
RACDCERT ID(WEBUSER) MAP WITHLABEL('INTERNET OTHERS') TRUST
         IDNFILTER('OU=VeriSign Class 1 Individual
                   Subscriber.O=VeriSign, Inc.L=Internet')
RACDCERT ID(NYUSER)  MAP WITHLABEL('NY OTHERS') TRUST
         SDNFILTER('OU=New York.OU=US.O=World Sales Corp')
RACDCERT ID(NYADMIN) MAP WITHLABEL('NY ADMIN') TRUST
         SDNFILTER('OU=Admin.OU=New York.OU=US.O=World Sales Corp')
         IDNFILTER('OU=VeriSign Class 1 Individual
                   Subscriber.O=VeriSign, Inc.L=Internet')
SETROPTS RACLIST(DIGTNMAP) REFRESH
```

Figure 64. Sample RACDCERT MAP commands not using a Model Certificate

Excluding a Certificate Using the NOTRUST Option

You can use certificate name filtering to prevent a digital certificate from being associated with a user ID by defining a name filter with the NOTRUST option, as long as it is the most specific filter that matches the certificate you wish to exclude. Certificate name filters defined with the NOTRUST option are not used to associate a user ID to a certificate. The NOTRUST option can be used to exclude one or more certificates.

The RACDCERT MAP command in Figure 65 defines a fully distinguished subject's and issuer's name filter labeled `'NOT FRANS'` with the NOTRUST option to prevent a certificate from being mapped to the `NYADMIN` user ID.

```
RACDCERT ID(NYADMIN) MAP WITHLABEL('NOT FRANS') NOTRUST
         SDNFILTER('CN=Frans De Graaff.OU=Admin.OU=New
                   York.OU=US.O=World Sales Corp')
         IDNFILTER('OU=VeriSign Class 1 Individual
                   Subscriber.O=VeriSign, Inc.L=Internet')
SETROPTS RACLIST(DIGTNMAP) REFRESH
```

Figure 65. Sample RACDCERT MAP command using the NOTRUST Option

Mapping Multiple User IDs Using Additional Criteria

You may need to assign more than one user ID to a certificate, based on the particular circumstances in which the certificate is presented. Such circumstances may include the following:

- The user of the certificate needs access to more than one application, and each application requires a different user ID.
- The same application may run on more than one system, and each system requires a different user ID.

Certificate name filtering allows you to associate more than one user ID to a certificate using additional criteria, such as APPLID and SYSID. Other criteria, such as SSL encryption level, may be used if this information passed with the certificate

by the caller of the initACEE callable service. For information about passing additional criteria to initACEE, see *z/OS SecureWay Security Server RACF Callable Services*.

You specify multiple user IDs for a filter using the RACDCERT MAP command with the MULTIID option, and creating one general resource profile in the DIGTCRIT class for each user ID you wish to associate with the filter. The name of the DIGTCRIT profile consists of one or more criteria values. The user ID is specified as the APPLDATA value. When you use RACDCERT MAP with the MULTIID option, you do not specify a user ID. Instead, you use the CRITERIA option of RACDCERT MAP to specify one or more variable names that correspond to values in the DIGTCRIT profile names. Therefore, each MULTIID filter is associated with profiles in the DIGTCRIT class instead of a user ID.

Using Application Criteria

When users have only one certificate but need to connect to multiple applications that require different user IDs, you can assign user IDs based on the application identifier (APPLID).

Example: Michael's Music Company has two web-based applications: an on-line royalties application, and an on-line inventory application. The company has contracted VeriSign to issue certificates to its users, one certificate for each user. When one of the company's users connects to the royalties application, the user's certificate should be assigned the ROYALID user ID. When one of the company's users connects to the inventory application, the user's certificate should be assigned the INVID user ID.

The RACDCERT MAP and RDEFINE commands shown in Figure 66 create a full issuer's name filter that maps these two user IDs based on the application being accessed by the user of the certificate. The RACDCERT command uses the MULTIID option to specify additional criteria contained in the DIGTCRIT class using the predefined variable &APPLID. The RDEFINE commands create two profiles in the DIGTCRIT class that associate each APPLID value with the user ID indicated by the APPLDATA value.

```
RACDCERT MULTIID MAP WITHLABEL('All Michael's Music Employees') TRUST
         IDNFILTER('OU=Michael's Music General Subscriber.O=VeriSign,
                   Inc.L=Internet')
         CRITERIA(APPLID=&APPLID)
SETROPTS RACLIST(DIGTNMAP) REFRESH

RDEFINE  DIGTCRIT APPLID=EROYAL APPLDATA(ROYALID)
RDEFINE  DIGTCRIT APPLID=EINV APPLDATA(INVID)
SETROPTS RACLIST(DIGTCRIT) REFRESH
```

Figure 66. Sample RACDCERT MAP and RDEFINE Commands for Mapping Multiple User IDs

You can display mapping information for a MULTIID filter using the RACDCERT LISTMAP command with the LABEL option. For example:

```
RACDCERT MULTIID LISTMAP(LABEL('All Michael's Music Employees'))
```

Figure 67 on page 542 shows sample output based on this RACDCERT LISTMAP command.

```
Mapping information for MULTIID:
 Label: All Michael's Music Employees
 Status: TRUST
 Issuer's Name Filter:
      >OU=Michael's Music General Subscriber.O=VeriSign, Inc.L=Internet<
 Subject's Name Filter:
      ><
 Criteria:
      APPLID=&APPLID
```

Figure 67. Sample Output from the LISTMAP Command for a MULTIID Filter

For details about using the RACDCERT MAP command with the MULTIID option, RACDCERT LISTMAP, and the RDEFINE command, see *z/OS SecureWay Security Server RACF Command Language Reference*.

If a user certificate is used for additional applications and should be associated with a user ID for these applications, you can create a generic DIGTCRIT profile named `APPLID=*` to cover all other applications. For example, the addition of the following DIGTCRIT profile to the MULTIID filter created in Figure 66 on page 541 specifies that the `ALLAPPS` user ID should be associated with all certificates used to access all other applications.

```
SETROPTS GENERIC(DIGTCRIT)
RDEFINE  DIGTCRIT APPLID=* APPLDATA(ALLAPPS)
SETROPTS RACLIST(DIGTCRIT) REFRESH
```

Note: If the caller of the `initACEE` callable service does not specify the APPLID variable, only the `APPLID=*` profile in the DIGTCRIT class will be used to determine the RACF user ID.

Using System Criteria

When users have only one certificate but need to connect to multiple systems that require different user IDs, you can assign user IDs based on the system identifier (SYSID). The SYSID is the 4-character SID value specified in the SMFPRMxx member of SYS1.PARMLIB on each system. You specify system criteria using the predefined variable `&SYSID` with the CRITERIA option of the RACDCERT MAP command for MULTIID filters. You must use the RDEFINE command to create profiles in the DIGTCRIT class to associate each SYSID value with the user ID indicated by the APPLDATA value.

Using Multiple Criteria

You can use multiple additional criteria with your certificate name filters by specifying multiple values with the CRITERIA option of the RACDCERT MAP command.

Example: Jamal's Bank has contracted with VeriSign to provide certificates to its customers and its account representatives. Both customers and account representatives access the company's systems through SSL. Customer SSL connections go through system A (SYSID=`SYSA`) and are only allowed access to general information about the company's offerings. Account representatives connect through system B (SYSID=`SYSB`) and need access to confidential customer information. Both systems A and B share the RACF database.

The application that serves the company's data invokes `initACEE` and passes user certificates with information about the SSL encryption level used by each user to connect to the system. This information is passed to `initACEE` as a variable called

ENCRLVL, and the following values are assigned by the application based on the SSL encryption strength of the connection:

HIGH SSL encryption strength using at least 128-bit encryption

LOW SSL encryption strength using 40-bit encryption

The RACDCERT MAP and DIGTCRIT commands shown in Figure 68 set up an issuer's name filter that uses multiple user IDs based on SYSID and ENCRLVL. In this example, there is a certificate available for use as a model in data set 'JAMALDC'. The certificate contains the following issuer's name.

```
OU=Jamal's Bank General Subscriber.O=VeriSign, Inc.L=Internet
```

```
RACDCERT MULTIID MAP('JAMALDC') WITHLABEL('All Jamal's Users')
         IDNFILTER('OU') CRITERIA(SYSID=&SYSID.ENCRLVL=&ENCRLVL)
SETROPTS RACLIST(DIGTNMAP) REFRESH

SETROPTS GENERIC(DIGTCRIT)
RDEFINE  DIGTCRIT SYSID=SYSB.ENCRLVL=HIGH APPLDATA('ACCTREP')
RDEFINE  DIGTCRIT SYSID=SYSB.ENCRLVL=*    APPLDATA('GENERAL')
RDEFINE  DIGTCRIT SYSID=SYSA.ENCRLVL=*    APPLDATA('GENERAL')
SETROPTS RACLIST(DIGTCRIT) REFRESH
```

Figure 68. Sample RACDCERT MAP and RDEFINE Commands using Multiple Criteria

The issuer's name filter created in Figure 68 associates the following user IDs:

GENERAL For all customers, and account representatives connecting with low-strength encryption.

ACCTREP For account representatives connecting with high-strength encryption.

Details for Processing an Issuer's Name Filter with Multiple Criteria: For example, if a customer accesses the Jamal's Bank system using an unregistered user certificate, the following represents the sequence of processing that RACF, specifically the `initACEE` callable service, will complete to process multiple criteria using a DIGTCRIT profile.

1. The sequence shown in "How RACF Processes Certificate Name Filters" on page 538 is followed, until the full issuer's name is used to check for a matching profile in the DIGTNMAP class, to determine if there is an applicable certificate name filter.

   Result: A DIGTNMAP profile is found to match:

   ```
   OU=Jamal's Bank General Subscriber.O=VeriSign, Inc.L=Internet
   ```

2. The criteria definitions, `SYSID=&SYSID.ENCRLVL=&ENCRLVL` are found in the DIGTNMAP profile, and the supplied values are substituted for each variable: `SYSID=SYSA` and `ENCRLVL=LOW`.

   Result: A DIGTCRIT profile is found to match:

   ```
   SYSID=SYSA.ENCRLVL=*
   ```

3. Processing by `initACEE` continues using the user ID `GENERAL` for the customer's certificate.

   Note: In this example, if the application calling the `initACEE` callable service does not pass the ENCRLVL variable, only the `SYSID=` value is used to

determine the user ID. Therefore, the DIGTCRIT profile named SYSID=SYSA.ENCRLVL=* is found to match, and the user ID GENERAL is still used for the customer's certificate.

Activating Additional Criteria

When you create a certificate name filter using the MULTIID option of the RACDCERT MAP command, you must create corresponding profiles in the DIGTCRIT class to specify the multiple user IDs associated with the filter. The DIGTCRIT class must be active and SETROPTS RACLIST processing must be active for the DIGTCRIT class. Before creating any profiles in the DIGTCRIT class, you must issue the following command:

```
SETROPTS CLASSACT(DIGTCRIT) RACLIST(DIGTCRIT)
```

Once SETROPTS RACLIST processing is active for the DIGTCRIT class, you must refresh the DIGTCRIT class in order for new or changed profiles to take effect. After creating or changing a DIGTCRIT class profile, you must issue the following command:

```
SETROPTS RACLIST(DIGTCRIT) REFRESH
```

Controlling Applications That Invoke the initACEE Callable Service

Authorized applications, such as servers, that invoke the initACEE callable service (IRRSIA00) can administer certificates associated with users and certificates associated with certificate authorities. You authorize these applications by administering the same FACILITY class resources checked by the RACDCERT command. See "Controlling the Use of the RACDCERT Command" on page 522.

Authorized applications, such as web servers, can also present a client's certificate containing a hostIdMappings extension and invoke the initACEE callable service to request to have a security context (ACEE) created or have the client's user ID queried and returned. You authorize these applications by administering profiles in the SERVAUTH class.

Registering User Certificates

Applications can invoke the initACEE callable service (IRRSIA00) and pass a user certificate requesting registration. If the caller's user ID has at least READ authority to the IRR.DIGTCERT.ADD resource, the certificate is associated with that user ID. This initACEE register function performs the same function as the RACDCERT ADD command. Therefore, a profile is created in the DIGTCERT class with the user ID in the APPLDATA field, and the user's profile is updated with the name of the DIGTCERT profile. Once the certificate is registered, it can be used in the same manner as a certificate that was registered using the RACDCERT command.

Note: RACF generates a label name for user certificates registered through the initACEE callable service. The generated label name is of the form LABEL*nnnnnnnn* where *nnnnnnnn* is the first integer value (starting at 00000001) that generates a unique label name.

Deregistering User Certificates

Applications can invoke the initACEE callable service (IRRSIA00) requesting deregistration of a user certificate. This deregistration function performs the same function as the RACDCERT DELETE command, and disassociates the certificate from the current user ID. If the caller's user ID has at least READ authority to the

IRR.DIGTCERT.DELETE resource, the profile in the DIGTCERT class associating the certificate with this user ID is deleted, and the name of the DIGTCERT profile is removed from the user's profile.

Replacing Certificate-Authority Certificates

Applications can invoke the `initACEE` callable service (IRRSIA00) and pass a certificate-authority certificate, requesting replacement of a previously registered certificate-authority certificate. If the caller's user ID has at least CONTROL authority to the IRR.DIGTCERT.ADD resource and the previously registered certificate-authority certificate is eligible for replacement, the certificate will be replaced and the new certificate will be associated with the `irrcerta` user ID.

A previously registered certificate-authority certificate is eligible for replacement when:

1. Its public key matches that of the input certificate-authority certificate.
2. Its subject's distinguished name matches that of the input certificate-authority certificate.
3. It has a private key.

If the caller has CONTROL authority to the IRR.DIGTCERT.ADD resource but the previously registered certificate-authority certificate is not eligible for replacement, it will not be replaced. The input certificate will be added as a user certificate and will be associated with the user ID of the caller. See "Registering User Certificates" on page 544.

Using a hostIdMappings Extension

Authorized applications, such as web servers, can present a client's certificate containing a `hostIdMappings` extension and invoke the `initACEE` callable service (IRRSIA00) to request to have a security context (ACEE) created or have the client's user ID queried and returned. For the format of the `hostIdMappings` extension, see *z/OS SecureWay Security Server RACF Callable Services*.

In these cases, the application is seeking to complete a login for a client whose certificate includes a `hostIdMappings` extension that may specify the user ID to be used on a particular server (host). Controlling an identity used for login purposes is a very important security objective. Therefore, you should exercise administrative control in the following areas by authorizing:

1. Which certificates with a `hostIdMappings` extension will be honored
2. Which servers will be authorized to accept logins using certificates that contain explicit user IDs and host names

When an application calls the `initACEE` callable service for this purpose and passes a certificate that has a `hostIdMappings` extension, the caller must have READ authority for the IRR.HOST.*host-name* resource defined in the SERVAUTH class, and the certificate must have been issued by a certificate authority that is defined with the HIGHTRUST option.

The `initACEE` callable service builds a security context (ACEE) for the user ID contained in `hostIdMappings` extension only if the certificate presented is not registered in the RACF database, and there is no matching certificate name filter.

Administering Profiles in the SERVAUTH Class

You authorize servers to accept logins for clients whose certificates contain a `hostIdMappings` extension by administering profiles in the SERVAUTH class. Be

sure that each server you wish to authorize is defined as a RACF user, if not already defined. Servers may run as jobs or started procedures. For example:

```
ADDGROUP WEBSRVGP
ADDUSER  WEBSRV1 GROUP(WEBSRVGP) NOPASSWORD
ADDUSER  WEBSRV2 GROUP(WEBSRVGP) NOPASSWORD
```

Note: You should assign protected user IDs for servers using the NOPASSWORD option. See “Assigning RACF User IDs to Started Procedures” on page 143.

Define resources in the SERVAUTH class using the following format:

```
IRR.HOST.host-name
```

Permit servers to access this resource with READ authority. This will allow them to accept logins for the host name specified in the resource name. For example, to allow the servers in the `WEBSRVGP` to accept logins for the host system called `MVSDSN1`, execute the following commands:

```
RDEFINE  SERVAUTH IRR.HOST.MVSDSN1 UACC(NONE)
PERMIT   IRR.HOST.MVSDSN1 CLASS(SERVAUTH) ID(WEBSRVGP) ACCESS(READ)
SETROPTS CLASSACT(SERVAUTH)
```

In this example, if a server running under the authority of user ID `WEBSRV1` presents a client certificate issued by a certificate authority with HIGHTRUST status and the certificate contains a `hostIdMappings` extension that includes a user ID mapping for host name `MVSDSN1`, a security context (ACEE) will be created for the user ID mapped to `MVSDSN1`, as indicated in the `hostIdMappings` extension.

Using the HIGHTRUST Option

You can control which certificates with a `hostIdMappings` extension will be honored by authorized servers. You do this by defining the certificate authority that issues these certificates as highly trusted on your system. For example:

```
RACDCERT CERTAUTH ALTER(LABEL('Local Certificate Authority')) HIGHTRUST
```

See *z/OS SecureWay Security Server RACF Command Language Reference* for details about syntax and authorization required for using the RACDCERT command.

Controlling Applications That Invoke the R_datalib Callable Service

Authorized applications, such as servers, that invoke the `R_datalib` callable service (IRRSDL00) can extract private keys and manage certificate serial numbers. You authorize these applications for these functions by administering the same FACILITY class resources checked by the RACDCERT command. See “Controlling the Use of the RACDCERT Command” on page 522.

Extracting Private Keys

Applications can invoke the `R_datalib` callable service (IRRSDL00) to extract the private keys from certain certificates.

User Certificates

An application can extract the private key from a user certificate if the following conditions are met:

1. The caller’s user ID is the user ID associated with the certificate.
2. The certificate is connected to its key ring with the PERSONAL usage option.

CERTAUTH and SITE Certificates

An application can extract the private key of a CERTAUTH or SITE certificate if the following conditions are met:

1. The caller's user ID has at least CONTROL authority to the resource IRR.DIGTCERT.GENCERT in the FACILITY class.
2. The certificate is connected to its key ring with the PERSONAL usage option.

Managing Certificate Serial Numbers

Applications can invoke the `R_datalib` callable service (IRRSDL00) to manage serial numbers for certain certificates.

User Certificates

An application can increment the ″last serial number issued″ (CERTLSER) for a user certificate if the following conditions are met:

1. The caller's user ID is the user ID associated with the certificate.
2. The caller's user ID has at least READ authority to the resource IRR.DIGTCERT.GENCERT in the FACILITY class.

CERTAUTH and SITE Certificates

An application can increment the ″last serial number issued″ (CERTLSER) for a CERTAUTH or SITE certificate, if the caller's user ID has at least CONTROL authority to the resource IRR.DIGTCERT.GENCERT in the FACILITY class.

Controlling Applications That Invoke the R_PKIServ Callable Service

Authorized applications, such as web servers, and their clients can invoke the `R_PKIServ` callable service (IRRSPX00) to request the generation and retrieval of X.509 V.3 certificates. You authorize these applications and clients by administering FACILITY class resources called IRR.RPKISERV.*function*, and, in certain cases, by also administering FACILITY class resources called IRR.DIGTCERT.*function*.

The function name in the IRR.RPKISERV.*function* resource name is either GENCERT or EXPORT. Therefore, the resource names are IRR.RPKISERV.GENCERT or IRR.RPKISERV.EXPORT. These functions perform the same functions as the RACDCERT GENCERT and RACDCERT EXPORT commands. Therefore, certificates processed by IRRSPX00 can be used in the same manner as those processed by the RACDCERT command.

Authorizing Servers and Clients

Servers and, in some cases, clients must be authorized to invoke IRRSPX00 to generate or retrieve X.509 user certificates. The server's user ID must have at least READ authority to IRR.RPKISERV.*function* to successfully invoke IRRSPX00. If the server's user ID has sufficiently high authority, the client may be able to successfully invoke IRRSPX00 without being specifically authorized.

If the server's user ID has READ authority to IRR.RPKISERV.*function*, then the client's user ID, if available, must have sufficient authority to the appropriate IRR.DIGTCERT.*function* resources. If the client's user ID is not available, then the server's user ID must have sufficient authority to the appropriate IRR.DIGTCERT.*function* resources. The authorization checks for the IRR.DIGTCERT.*function* resources that are performed using the client's or server's user ID are similar to the checks performed for the RACDCERT GENCERT and EXPORT commands. See *z/OS SecureWay Security Server RACF Command Language Reference* for detailed information about authorities required to execute the RACDCERT GENCERT and EXPORT commands.

If the server's user ID has UPDATE (or higher) authority to IRR.RPKISERV.*function*, the client's user ID needs no specific authority to IRR.RPKISERV.*function* or IRR.DIGTCERT.*function* resources.

If the server's user ID has UPDATE authority to IRR.RPKISERV.*function*, then the server's user ID must have sufficient authority to the appropriate IRR.DIGTCERT.*function* resources.

If the server's user ID has CONTROL (or higher) authority to IRR.RPKISERV.*function*, or the server's user ID has SPECIAL authority, then the server's user ID does not need any authority to the IRR.DIGTCERT.*function* resources.

Automatic Registration of Digital Certificates

Your installation can provide a user interface to allow users to register their own digital certificates. You can provide an HTML web page and CGI program accessed through WebSphere Application Server. (See the sample provided in 'SYS1.SAMPLIB' member RACINSTL.) The registration page can be used to prompt for registration of the user's certificate for his or her RACF user ID. When the user clicks on the registration box, a secure session is set up using SSL and the user's digital certificate. The user is prompted for his or her RACF user ID and password, which is passed from WebSphere Application Server to z/OS UNIX, then to RACF through the `initACEE` callable service (IRRSIA00) for registration. RACF verifies the user ID and password and creates an ACEE. Note that because the validity of the certificate is established when the SSL connection is set up, the DIGTCERT profile for this certificate is marked TRUSTed.

See "Registering User Certificates" on page 544 for details about using the registration function of the `initACEE` callable service.

Integrated Cryptographic Service Facility (ICSF) Considerations

Integrated Cryptographic Service Facility (ICSF) is a software element of z/OS and OS/390 that provides the application programming interfaces to the cryptographic hardware. ICSF is recommended for the storage of the private keys associated with digital certificates. ICSF is a more secure solution than non-ICSF private key management. ICSF ensures that private keys are encrypted under the ICSF master key and that access to them is controlled by RACF general resources in the CSFKEYS and CSFSERV classes. In addition, operational performance is improved because ICSF utilizes the hardware CMOS Cryptographic Coprocessor.

If ICSF is implemented at your installation, you can specify its use for storage of private keys by using the ICSF operand of the RACDCERT GENCERT and RACDCERT ADD commands. For information about specifying ICSF with the RACDCERT command, see *z/OS SecureWay Security Server RACF Command Language Reference*.

You can migrate non-ICSF private keys to ICSF simply by issuing the RACDCERT ADD command. Use the ICSF operand and specify the data set containing the existing certificate. If the certificate data set is no longer available, it may be recreated using the RACDCERT EXPORT command.

The irrcerta, irrmulti, and irrsitec User IDs

The irrcerta, irrmulti, and irrsitec user IDs are defined in USER profiles that are supplied with RACF and can not be defined by your installation. They are used to anchor certain profiles in the DIGTCERT and DIGTNMAP class that are not associated with individual user IDs, and cannot be used for any other purpose.

- User certificates that you add using the RACDCERT ADD command with the CERTAUTH option are automatically associated with the user ID irrcerta.
- User certificates that you add using the RACDCERT ADD command with the SITE option are automatically associated with the RACF user ID irrsitec.
- Certificate name filters that you add using the RACDCERT MAP command with the MULTIID option are automatically associated with the RACF user ID irrmulti.

The use of these user IDs in DIGTCERT and DIGTNMAP profiles is automatic and cannot be changed using RACF commands. These user IDs cannot be administered using the ADDUSER, ALTUSER, DELUSER, LISTUSER and CONNECT commands. Since profiles that are associated with these user IDs contain lowercase characters, the SEARCH FILTER command is not intended for use in locating them and will deliver unpredictable results.

The irrcerta, irrmulti, and irrsitec user ID profiles should not be deleted. They must exist at initialization time or RACF initialization will automatically add them. In addition, the remove ID utility (IRRRID00) will not create commands to process these user IDs. For more information about IRRRID00 and the processing of DIGTCERT and DIGTNMAP profiles, see “Finding Residual IDs” on page 347.

Implementation Scenarios

Scenario 1: Secure Server with a Certificate Signed by a Certificate Authority

Secure servers require the ability to retrieve the certificate that is associated with a particular server, along with the ability to perform operations with the private key of the server, such as establishing an SSL session. Assume that we have a secure server which has the distinguished name of C=US,O=XYZZY OU=Inventory, and a domain name of xyzzy.com. This server executes on z/OS with the user ID INVSERV. The steps to implement a server certificate are:

1. Generate a self-signed certificate for the server. This certificate is associated with the user ID that is associated with the secure server.

```
RACDCERT ID(INVSERV)
         GENCERT
         SUBJECTSDN(CN('xyzzy.com')
                    OU('Inventory')
                    O('XYZZY')
                    C('US'))
         WITHLABEL('Inventory Server')
```

   Note: SSL requires that the domain name must be the common name (CN).

2. Create a certificate request to send to our chosen certificate authority. The certificate request that we are creating is based on the certificate that we created in the step above. Place this certificate into the data set ’MARKN.INVSERV.GENREQ’.

```
RACDCERT ID(INVSERV)
         GENREQ(LABEL('Inventory Server'))
         DSN('MARKN.INVSERV.GENREQ')
```

3. Send the certificate request to the certificate authority. The certificate request is in base64-encoded text. Typically, the request is sent to the certificate authority by using ″cut and paste″ to place the certificate request into an e-mail that is sent to the certificate authority.

   Note: RACF is not involved with this step.

4. The certificate authority validates the certificate. If the certificate is approved by the certificate authority, it is signed by the certificate authority, and returned to the requestor.

   Note: RACF is not involved with this step.

5. Receive the returned certificate into a data set (e.g. 'MARKN.INVSERV.CERT'). The returned certificate is in base64-encoded text. This may be done with ″cut and paste″, FTP, or other technique.

   Note: RACF is not involved with this step.

6. Replace the self-signed certificate with the certificate signed by the certificate authority. Note that the certificate is only replaced if the user ID that is specified as the ID value on the RACDCERT ADD command is the same user ID that was specified when the certificate was created. If the ID is not the same, then the certificate is added anew.

```
RACDCERT ID(INVSERV)
         ADD('MARKN.INVSERV.CERT')
         WITHLABEL('Inventory Server')
```

7. Connect the certificate to INVSERV's existing key ring and mark it as the default certificate.

```
RACDCERT ID(INVSERV)
         CONNECT(LABEL('Inventory Server')
         RING(RING01)
         DEFAULT)
```

Scenario 2: Secure Server with a Locally Signed Certificate

This is similar to “Scenario 1: Secure Server with a Certificate Signed by a Certificate Authority” on page 549 with the exception that the certificate assigned to the secure server is a locally signed certificate rather than one signed by a certificate authority. Assume that the local certificate authority has the distinguished name of `C=US,O=XYZZY OU='Local Certificate Authority'.` The steps to implement a locally signed server certificate are:

1. Generate a self-signed certificate to represent the local certificate authority. This certificate is used as the certificate-authority certificate.

```
RACDCERT CERTAUTH
         GENCERT
         SUBJECTSDN(OU('Local Certificate Authority')
                    O('XYZZY')
                    C('US'))
         KEYUSAGE(CERTSIGN)
         WITHLABEL('XYZZY Local Certificate Authority')
```

2. Export the certificate to a data set, in this case 'MARKN.LOCCERTA.CERT'.

```
RACDCERT CERTAUTH
         EXPORT(LABEL('XYZZY Local Certificate Authority'))
         DSN('MARKN.LOCCERTA.CERT')
```

3. Place the certificate into the z/OS UNIX Hierarchical File System (HFS).

```
OPUT 'MARKN.LOCCERTA.CERT' '/u/loccerta/certauth.cacert'
```

   Note: RACF is not involved with this step.

4. Configure WebSphere Application Server to recognize the file `/u/loccerta/certauth.cacert` as a certificate-authority MIME type.

   Note: RACF is not involved with this step.

5. Each end user must point their browser to the HFS file containing the certificate and run an acceptance dialog to allow the browser to accept the self-signed certificate. Each browser has its own mechanism for performing this step.

   Note: RACF is not involved with this step.

6. Logon to the server user ID `INVSERV` and create a certificate for the server, signed with the certificate-authority certificate that was created in step 1 on page 550.

```
RACDCERT ID(INVSERV)
         GENCERT
         SUBJECTSDN(CN('xyzzy.com')
                    OU('Inventory')
                    O('XYZZY')
                    C('US'))
         WITHLABEL('Inventory Server')
         SIGNWITH(CERTAUTH
                 LABEL('XYZZY Local Certificate Authority'))
```

Scenario 3: Migrating an ikeyman Certificate

The installation needs to migrate their existing certificates on z/OS. These certificates were created with the `ikeyman` utility and reside in the z/OS UNIX HFS. The steps to migrate these certificates are:

1. Using `ikeyman`, export the certificate from the `ikeyman` key ring file as a PKCS#12 export file and place it into the z/OS UNIX HFS.

   Note: RACF is not involved with this step.

2. Export the file to an MVS data set, in this case 'MARKN.IMPORTED.CERT'.

```
OGET '/u/markn/cert.usercert' 'MARKN.IMPORTED.CERT'
```

3. Add the certificate to the RACF database and assign it a user. Assume that `ikeyman` encrypted the certificate with the password `xyz`.

```
RACDCERT ID(MARKN)
         ADD('MARKN.IMPORTED.CERT')
         WITHLABEL('Mark's Personal Certificate')
         TRUST
         PASSWORD('xyz')
```

Scenario 4: Secure Server-to-Server Session Enablement

A company wishes to use two different secure servers for two different applications. The first application is for its internal employee data, which allows employees to read their own information, and allows designated employees to modify information. This server is called `internal_ss`. The company also has an external secure server, which is used by client applications running on the customer's systems, to order materials and check the status of orders. This server is called `external_ss`. The `internal_ss` server executes with a user ID of INSS, and accepts certificates only from the company's internal certificate authority, whose name is ″ACME Local Certificate Authority,″ and whose certificate is located in the data set 'ACME.CERTIF'. The `external_ss` server executes with a user ID of EXSS, and accepts certificates from either the internal certificate authority ″ACME Local Certificate Authority″ or the external certificate authority called ″Really Big Certificate Authority.″ Really Big Certificate Authority's certificate is in the data set 'REALBIG.CERTIF'. The commands to accomplish this are shown below. The

authority checks that are shown assume that the person who issues these commands does not have SPECIAL authority, and is neither the user ID INSS or the user ID EXSS.

1. Create the user IDs for the secure servers.
   - ADDUSER INSS
   - ADDUSER EXSS
   - Authority required: CLAUTH for the USER class and JOIN in the default group to which they are connected.
2. Add the certificate-authority certificates.
   - `RACDCERT CERTAUTH ADD('ACME.CERTIF')    TRUST WITHLABEL('ACME')`
   - `RACDCERT CERTAUTH ADD('REALBIG.CERTIF') TRUST WITHLABEL('Really Big')`
   - Authority required: CONTROL to the IRR.DIGTCERT.ADD resource in the FACILITY class.
3. Add the server certificates and the associated private keys. On platforms other than z/OS or OS/390, this is performed using a facility such as `mkkf`, `ikeyman`, or equivalent. On z/OS or OS/390, this is performed using the RACDCERT GENCERT command. To generate the certificates for the two servers, these two RACDCERT commands are required:

   ```
   RACDCERT  ID(INSS)
             GENCERT
             SUBJECTSDN(CN('Internal Secure Server')
                        C('US'))
             WITHLABEL('INSS-001')
             SIGNWITH(CERTAUTH LABEL('ACME'))

   RACDCERT  ID(EXSS)
             GENCERT
             SUBJECTSDN(CN('External Secure Server')
                        C('US'))
             WITHLABEL('EXSS-001')
             SIGNWITH(CERTAUTH LABEL('Really Big'))
   ```

   Authority required: UPDATE to the resource IRR.DIGTCERT.ADD in the FACILITY class, along with CONTROL to the resource IRR.DIGTCERT.GENCERT in the FACILITY class.
4. Create key rings for the secure servers.
   - RACDCERT ID(EXSS) ADDRING(RING01)
   - RACDCERT ID(INSS) ADDRING(RING01)
   - Authority required: UPDATE to the resource IRR.DIGTCERT.ADDRING in the FACILITY class.
5. Connect the certificates to INSS's key ring.
   - RACDCERT ID(INSS) CONNECT(ID(INSS) LABEL('INSS-001') RING(RING01) DEFAULT)
   - RACDCERT ID(INSS) CONNECT(CERTAUTH LABEL('ACME') RING(RING01))
   - Authority required: CONTROL to the resource IRR.DIGTCERT.CONNECT in the FACILITY class.
6. Connect the certificates to EXSS's key ring.
   - RACDCERT ID(EXSS) CONNECT(ID(EXSS) LABEL('EXSS-001') RING(RING01) DEFAULT)
   - RACDCERT ID(EXSS) CONNECT(CERTAUTH LABEL('ACME') RING(RING01))

- RACDCERT ID(EXSS) CONNECT(CERTAUTH LABEL('Really Big') RING(RING01))
- Authority required: CONTROL to the resource IRR.DIGTCERT.CONNECT in the FACILITY class.

Scenario 5: Creating Client Browser Certificates with a Locally Signed Certificate

The installation wishes to locally issue client browser certificates. This is similar to "Scenario 2: Secure Server with a Locally Signed Certificate" on page 550 in that a local certificate-authority certificate must first be created. In this case, a client certificate is created, locally signed, exported from RACF in PKCS#12 format, and imported into the user's browser.

1. Follow steps 1 through 6 as described in "Scenario 2: Secure Server with a Locally Signed Certificate" on page 550 to create a local certificate-authority certificate to use for signing client browser certificates.
2. User MARKN can obtain a local browser certificate for himself using the following command:

```
RACDCERT ID(MARKN)
         GENCERT
         SUBJECTSDN(CN('Mark Napolitano')
                    OU('Local Certificate Authority')
                    O('XYZZY')
                    C('US'))
         WITHLABEL('My Browser Cert')
         KEYUSAGE(HANDSHAKE)
         SIGNWITH(CERTAUTH LABEL('XYZZY Local Certificate Authority'))
```

3. Export the certificate and private key to an MVS data set in PKCS#12 binary form where the password is 'The circus is coming':

```
RACDCERT ID(MARKN)
         EXPORT
         LABEL('My Browser Cert')
         DSN('MARKN.BROWSERC.P12BIN')
         PASSWORD('The circus is coming')
         FORMAT(PKCS12DER)
```

4. Use FTP to send the exported certificate data set in binary format to the target workstation. Use the appropriate browser-specific procedure to import the PKCS#12 package.

   Note: RACF is not involved with this step.
5. Optionally, the certificate labeled 'My Browser Cert' may be deleted from the RACF database if an appropriate certificate name filter is available to provide a user ID association, and the specific association between this certificate and the user ID MARKN is not required.

Chapter 20. RACF and SecureWay Network Authentication Service

This chapter provides information on using RACF with SecureWay Network Authentication Service.

Network Authentication Service uses RACF to store and administer information about principals and realms, using RACF user profiles and general resource profiles. The KERB segment of the user profile is used to store information about Network Authentication Service principals on your local system. The general resource class KERBLINK allows you to map principals to RACF user IDs on your system. The general resource class REALM defines the local Network Authentication Service realm and its trust relationships with foreign realms.

RACF also provides a callable service named `R_ticketserv` (IRRSPK00) for application servers that use Network Authentication Service services. See “Controlling Applications That Invoke the R_ticketserv Callable Service” on page 562 for more information.

This chapter describes how to use RACF to complete the following steps in the implementation of Network Authentication Service.

1. Customizing the local environment.
   a. Defining your local RRSF (RACF remote sharing facility) node.
   b. Defining your local realm.
   c. Defining local principals.
2. Defining your foreign environment.
   a. Defining foreign realms.
   b. Mapping RACF user IDs for foreign principals.

See the Network Authentication Service product documentation for implementation details.

Customizing your Local Environment

Before beginning to create RACF definitions to support your Network Authentication Service implementation, you must define your local system as the local RRSF node. If you have already implemented RRSF, you should review “RRSF Considerations for Network Authentication Service” on page 384 before beginning to create RACF definitions for principals and realms.

In this section, we will discuss defining your local server as the local realm. Once this definition is complete, you can begin defining your users as local principals and ensuring that their keys are registered with your local server.

Defining Your Local RRSF Node

You must define your local system as the local RRSF node using the TARGET command, even if you are not planning to exploit RRSF functions. For example:

```
@TARGET NODE(ENDMVSA) LOCAL
```

You must define your local system as the local RRSF node to allow keys to be generated for local principals who have their passwords changed through application updates. If the local RRSF node is not defined, RACF will not generate keys for local principals who change their own passwords. See “Generating Keys for Local Principals” on page 557 for more information.

If RRSF is already implemented, you may have already defined your local system as a local RRSF node. However, be sure to review “RRSF Considerations for Network Authentication Service” on page 384.

Defining Your Local Realm

You must define your local realm to RACF before you define local principals. This is because the local realm name is used to generate keys for local principals. You define your local realm by creating a profile in the REALM class called KERBDFLT. Using the KERB option of the RDEFINE and RALTER commands, you can specify the following information about your local realm:

KERBNAME Name of the local realm.

MINTKTLFE Minimum ticket lifetime for the local realm.

DEFTKTLFE Default ticket lifetime for the local realm.

MAXTKTLFE Maximum ticket lifetime for the local realm.

ENCRYPT Key encryption options (DES, DES3, DESD).

PASSWORD Value of the password for the local realm.

Notes:

1. This password is not a RACF user password. Therefore, it is not constrained by SETROPTS password rules that may be specified to control user passwords. In addition, the installation-defined new-password exit (ICHPWX01) is not invoked.
2. A password value must be supplied.
3. Uppercase and lowercase letters are accepted and maintained in the case in which they are entered.

See *z/OS SecureWay Security Server RACF Command Language Reference* for detailed information about using the KERB option of the RDEFINE and RALTER commands to administer profiles in the REALM class.

Example of defining the local realm

The following example shows a local realm KRB2000.IBM.COM being defined with a minimum ticket lifetime of 30 seconds, a default ticket lifetime of 10 hours, a maximum ticket lifetime of 24 hours, and a password of 744275. All of the ticket lifetimes are specified in seconds. The administrator then lists the new REALM profile.

```
RDEFINE REALM KERBDFLT KERB(KRB2000.IBM.COM) MINTKTLFE(30) +
        DEFTKTLFE(36000) MAXTKTLFE(86400) PASSWORD(NEW1pw)

RLIST REALM KERBDFLT KERB NORACF
 CLASS      NAME
 -----      ----
 REALM      KERBDFLT

 KERB INFORMATION
 ----------------
 KERBNAME=  KRB2000.IBM.COM
 MINTKTLFE= 0000000030
 MAXTKTLFE= 0000086400
 DEFTKTLFE= 0000036000
 KEY VERSION= 001
```

See *z/OS SecureWay Security Server RACF Command Language Reference* for RLIST authorization requirements.

Defining Local Principals

You must define local principals as RACF users using the ADDUSER and ALTUSER commands with the KERB option. This creates a KERB segment in the user profile. Each local principal must have a RACF password. Therefore, do not use the NOPASSWORD option when defining local principals.

You can specify the following information for your local principals:

KERBNAME Local principal name.

Note: Uppercase and lowercase letters are accepted and maintained in the case in which they are entered.

MAXTKTLFE Maximum ticket lifetime for the local principal.

ENCRYPT Key encryption options (DES, DES3, DESD).

See *z/OS SecureWay Security Server RACF Command Language Reference* for detailed information about using the KERB option of the ADDUSER and ALTUSER commands to administer user profiles for local principals.

For example:

```
ALTUSER LEMIEUX KERB(KERBNAME('JacquesLemieux'))
```

Generating Keys for Local Principals

Each local principal must have a key registered with the local Network Authentication Service server in order to be recognized as a local principal. The user's definition as a local principal is not complete until the key is generated. The key is generated from the principal's RACF user password at the time of the user's

password change. If you want a key to be generated, be sure to use a password change facility that will not result in an expired password that the user must change at next logon. For example, you can use the NOEXPIRED option of the ALTUSER command.

System Considerations for Key Generation: In order to successfully complete password changes for key generation, the RACF address space must be started and must be executing under the authority of a user ID that is associated with a user identifier (UID). In addition, each user or administrator who generates keys must also have an associated UID. As an alternative to defining individual UIDs for each user ID, you can customize your z/OS UNIX security environment to allow default OMVS segment processing. For more information, see "Defining User Identifiers (UIDs)" on page 504 and "Using Default OMVS Segments in USER and GROUP Profiles" on page 506.

SETROPTS KERBLVL processing: You can use the SETROPTS KERBLVL command to control the key types that can be used for authentication by users and realms on your RACF system. The supported key types are DES, DES3, and DESD (DES with derivation). They can be set using the KERB ENCRYPT operand of the ADDUSER, ALTUSER, RDEFINE and RALTER commands. The SETROPTS KERBLVL(0) setting allows only DES keys to be used for all users and realms. The SETROPTS KERBLVL(1) setting allows the DES, DES3, and DESD keys to be used, based on the options specified or defaulted in each user and realm profile. See *z/OS SecureWay Security Server RACF Command Language Reference* for more information about using the ADDUSER, ALTUSER, RDEFINE and RALTER commands to set KERB ENCRYPT options.

When KERBLVL(0) is in effect, only DES keys are generated at key generation time, and are used for authentication, regardless of the settings indicated in each user and realm profile. When KERBLVL(1) is effect, all three key types are generated at key generation time, regardless of the settings indicated in each user and realm profile; however, only the key types indicated in the profiles will be used. If no key types are indicated, no key will be available for authentication, and the user or realm will be unable to use the facilities of Network Authentication Service.

Once SETROPTS KERBLVL(1) has been enabled, it is inadvisable to reset to KERBLVL(0). This may create inconsistencies in the processing of encryption information for user and realm profiles. In these cases, the profiles may indicate that DES3 and DESD keys should be used, but they will not be used until KERBLVL(1) is enabled again.

Before enabling KERBLVL(1) for the first time, you may choose to specify the DES3 or DESD encryption options in advance for each user and realm while KERBLVL(0) is still in effect. The options will be ignored, but once KERBLVL(1) is enabled, these keys will be available for use, as soon as they are generated at next key generation time.

Coexistence considerations: If your installation shares the RACF database among systems that support KERB segments for user profiles and those that do not, you should not allow users to change their own passwords for key generation from systems that do not support the KERB segment. Keys will only be generated when users change their passwords from systems that support the KERB segment. Password changes made from systems that do not support the KERB segment will not generate keys to complete the users' local principal definitions.

If your installation shares the RACF database among systems that support the SETROPTS KERBLVL option and those that do not, you should not enable KERBLVL(1). This could allow the creation of keys that are unusable on downlevel systems. If KERBLVL(1) is desired, you should ensure that keys are not generated or authenticated on downlevel systems.

Methods for Generating Keys: Security administrator method: You can change a user's password so that a key can be generated using the ALTUSER command with the NOEXPIRED option. For example:

```
ALTUSER LEMIEUX PASSWORD(new1pw) NOEXPIRED
```

Notes:

1. Do not use the NOPASSWORD option.
2. You must specify a password value so a key can be generated.
3. All characters of the password will be folded to uppercase.

Alternatively, you can add a KERB segment for a local principal and generate the required key using a single RACF command. For example:

```
ALTUSER LEMIEUX PASSWORD(NEW1PW) NOEXPIRED KERB(KERBNAME('JacquesLemieux'))
```

User method: Users can change their own passwords, completing their own definitions as local principals, by using any standard RACF password-change facility, such as one of the following:

- TSO PASSWORD command (without the ID option)
- TSO logon
- CICS signon

Notes:

1. Password change requests from applications that encrypt the password prior to calling RACF will not result in usable keys.
2. All characters of the password will be folded to uppercase.

A local principal's key will be revoked whenever the user's RACF user ID is revoked or the RACF password is considered expired. If the user's key is revoked, the server will reject ticket requests from this user.

Automatic Local Principal Name Mapping

For each local principal you define on your system using the KERB option of the ADDUSER and ALTUSER commands, RACF automatically creates a mapping profile in the KERBLINK class. When you issue the ALTUSER command with the NOKERB option or issue a DELUSER for a user with a KERB segment, RACF automatically deletes the KERBLINK profile.

The KERBLINK profile maps the local principal name to the user's RACF user ID. The name of the KERBLINK profile for a local principal is the principal name specified as the KERBNAME value with the ADDUSER or ALTUSER command.

System Considerations for KERBLINK Profiles

Your installation may share the RACF database among systems that support KERBLINK class profiles and those that do not. However, you should not administer user profiles containing KERB segments from systems that do not support the KERBLINK class profiles.

Attention

Do not execute the DELUSER command, or an ALTUSER command with the NOKERB option, for a user profile that contains a KERB segment from RACF systems that do not support the KERBLINK class. These systems do not automatically manage KERBLINK profiles. You will inadvertently leave residual mapping profiles in the KERBLINK class. For information about recovery procedures, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

Considerations for Local Principal Names

The name of the KERBLINK profile contains the local principal name that is being mapped. Local principal names may contain imbedded blanks and lower case characters.

Blanks are not permitted as a part of a RACF profile name. Therefore, when building the KERBLINK profile name, as a result of specifying KERBNAME with the ADDUSER or ALTUSER command, RACF command processing will replace each blank with the X'4A' character (which often resolves to the ¢ symbol). This can be seen in the output from the `RLIST KERBLINK *` command, and in the output from the RACF data base unload utility (IRRDBU00). RACF command processing also prevents the X'4A' character (¢) from being specified as part of the actual local principal name.

You should use caution when specifying embedded blanks and lower case characters in local principal names if:

- Your installation shares the RACF database among systems that support the KERBLINK class and systems that do not.
- Administrators may issue DELUSER commands, or ALTUSER commands with the NOKERB option, from systems that do not support the KERBLINK class.

Residual KERBLINK profiles will be left if these commands are issued from systems that do not support the KERBLINK class. If the profile name contains blanks or lower case characters, you may be unable to remove these profiles using RACF commands. For information about recovery procedures, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

For information about the set of characters supported for local principal names, see *z/OS SecureWay Security Server RACF Command Language Reference*.

Customizing your Foreign Environment

Your local Network Authentication Service server can trust authentications completed by other servers, and can be trusted by other servers, by participating in trust relationships. See the Network Authentication Service product documentation for details about trust relationships.

To participate in trust relationships, you must define each server as a foreign realm. Then, you can allow users who are authenticated in foreign realms (foreign principals) to access protected resources on your local system by mapping one or more RACF user IDs to foreign principal names. You do not need provide foreign principals ability to logon to your local system. You can simply provide mapping to one or more local user IDs so they can gain access privileges for local resources that are under the control of an application server, such as DB2.

Defining Foreign Realms

You define foreign realms by creating profiles in the REALM class. The profile name of the REALM class profile contains the fully qualified name of both servers in the relationship. The profile name uses the following formats:

`/.../realm_1/KRBTGT/realm_2`

Using the KERB option of the RDEFINE and RALTER commands, you define a REALM profile and specify the following information:

PASSWORD
Value of the password for this trust relationship with a foreign realm.

Notes:

1. This password is not the same as a RACF user password. Therefore, it is not constrained by the SETROPTS password rules that may be specified to control user passwords.
2. A value must be supplied to establish a trust relationship with this foreign realm.

See *z/OS SecureWay Security Server RACF Command Language Reference* for detailed information about using the KERB option of the RDEFINE and RALTER commands to administer profiles in the REALM class.

For example:

```
RDEFINE REALM /.../KERBZOS.ENDICOTT.IBM.COM/KRBTGT/KER2000.ENDICOTT.IBM.COM +
        KERB(PASSWORD(1276458))
```

Mapping Foreign Principal Names

You map foreign principals names to RACF user IDs on your local system by defining general resource profiles in the KERBLINK class. You can map each principal in a foreign realm to its own user ID on your local system, or you can map all principals in a foreign realm to the same user ID on your system.

RACF user IDs that map to foreign principals do not need KERB segments. These user IDs are intended to be used only to provide local RACF identities to associate with access privileges for local resources that are under the control of an application server, such as DB2.

Each mapping profile in the KERBLINK class is defined and modified using the RDEFINE and RALTER commands. The name of the KERBLINK profile for a foreign principal contains the principal name, fully qualified with the name of the foreign realm. The profile name uses the following format:

`/.../foreign_realm/[foreign-principal_name]`

If you wish to map a unique RACF user ID to each foreign principal, you must specify the foreign realm name and the foreign principal name. If you wish to map the same RACF user ID to every foreign principal in the foreign realm, you need only specify the foreign realm name. In each case, you specify the local user ID using the APPLDATA option of the RDEFINE or RALTER command.

Example of mapping foreign principal names

In the following example, the users `SYKORA` and `NEDVED` will have their foreign principal names mapped with individual user IDs on the local z/OS system. All other

foreign principals presenting tickets from the KERBZOS.ENDICOTT.IBM.COM server will be mapped to the ENDKERB user ID on the local z/OS system.

```
RDEFINE KERBLINK /.../KERBZOS.ENDICOTT.IBM.COM/SYKORA APPLDATA('PETRS')
RDEFINE KERBLINK /.../KERBZOS.ENDICOTT.IBM.COM/NEDVED APPLDATA('PAVELN')
RDEFINE KERBLINK /.../KERBZOS.ENDICOTT.IBM.COM/       APPLDATA('ENDKERB')
```

Note: All characters of the foreign realm name and the foreign principal name will be folded to uppercase.

See *z/OS SecureWay Security Server RACF Command Language Reference* for detailed information about using the RDEFINE and RALTER commands to administer mapping profiles for foreign principals in the KERBLINK class.

Controlling Applications That Invoke the R_ticketserv Callable Service

Authorized applications, such as servers, can invoke the R_ticketserv callable service (IRRSPK00) to extract principal names from a GSS-API context token. This enables an application server to determine the client principal who originated an application-specific request, when the request includes a GSS-API context token and the intended recipient is the application server. For detailed information about invoking the R_ticketserv callable service, see *z/OS SecureWay Security Server RACF Callable Services*.

Applications that run in system key or in supervisor state do not require RACF authorization to use the R_ticketserv callable service. Applications that do not run in system key or in supervisor state require RACF authorization.

To authorize applications that do not run in system key or supervisor state, you must define RACF user IDs for them. These RACF user IDs must be given READ access to a RACF general resource called IRR.RTICKETSERV in the FACILITY class.

Defining Applications as RACF Users

Each application server must be defined as a RACF user, if not already defined. It may run as a job or a started procedure. If the application server executes as a batch job, the RACF user ID that is added is the user ID associated with the batch job. If the server executes as a started procedure, you must assign a RACF user ID using one of the following methods:

- Add the procedure name as an entry in the STARTED class. (This is the preferred method.)
- Add the procedure name in the RACF started procedure table (ICHRIN03), unless this table has already been modified by your installation to contain a generic entry.

In addition, you should assign the PROTECTED attribute to the user IDs that you associate with application servers. For more information, see “Assigning RACF User IDs to Started Procedures” on page 143.

Permitting Access to the IRR.RTICKETSERV Resource

Authorization to use the R_ticketserv callable service (IRRSPK00) is controlled through a RACF general resource called IRR.RTICKETSERV in the FACILITY class. You must define a profile to protect this resource, and then permit application user IDs to access the resource with READ authority.

Attention

Make sure an existing generic profile in the FACILITY class does not inadvertently grant this authority by default. It is recommended that you create a profile to protect the IRR.RTICKETSERV resource with UACC(NONE) until you determine which applications require access.

The following example protects the IRR.RTICKETSERV resource in the FACILITY class with UACC(NONE) and authorizes an application server called SERVER9 to use R_ticketserv.

```
RDEFINE FACILITY IRR.RTICKETSERV UACC(NONE)
PERMIT  IRR.TICKETSERV CLASS(FACILITY) ID(SERVER9) ACCESS(READ)
```

The FACILITY class must be active to enable applications to use R_ticketserv. If it is not already active at your installation, you must activate the FACILITY class using the SETROPTS command.

```
SETROPTS CLASSACT(FACILITY)
```

If your installation maintains FACILITY class profiles in storage through SETROPTS RACLIST processing, you must issue the following command to refresh the FACILITY class after you define or alter any profiles protecting the IRR.RTICKETSERV resource:

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

Appendix A. Description of RACF Classes

The following sections describe the general resource classes that are supplied in the class descriptor table (CDT). See *z/OS SecureWay Security Server RACF Macros and Interfaces* to find details (such as POSIT values) for each class.

Supplied Resource Classes for z/OS and OS/390 Systems

Table 39 lists the supplied classes that can be used on z/OS and OS/390 systems.

Table 39. Resource Classes for z/OS and OS/390 Systems

Class Name	Description
ALCSAUTH	Supports the Airline Control System/MVS (ALCS/MVS) product.
APPCLU	Verifying the identity of partner logical units during VTAM session establishment.
APPCPORT	Controlling which user IDs can access the system from a given LU (APPC port of entry). Also, conditional access to resources for users entering the system from a given LU.
APPCSERV	Controlling whether a program being run by a user can act as a server for a specific APPC transaction program (TP).
APPCSI	Controlling access to APPC side information files.
APPCTP	Controlling the use of APPC transaction programs.
APPL	Controlling access to applications.
CBIND	Controlling the client's ability to bind to the server.
CONSOLE	Controlling access to MCS consoles. Also, conditional access to other resources for commands originating from an MCS console.
CSFKEYS	Controlling use of Integrated Cryptographics Service Facility (ICSF) cryptographic keys. See also the GCSFKEYS class.
CSFSERV	Controlling use of Integrated Cryptographics Service Facility (ICSF) cryptographic services.
DASDVOL	DASD volumes. See also the GDASDVOL class.
DBNFORM	Reserved for future IBM use.
DEVICES	Used by MVS allocation to control who can allocate devices such as: • Unit record devices (printers and punches) (allocated only by PSF, JES2, or JES3) • Graphics devices (allocated only by VTAM) • Teleprocessing (TP) or communications devices (allocated only by VTAM)
DIGTCERT	Contains digital certificates and information related to them.
DIGTCRIT	Specifies additional criteria for certificate name filters.
DIGTNMAP	Mapping class for certificate name filters.
DIGTRING	Contains a profile for each key ring and provides information about the digital certificates that are part of each key ring.
DIRAUTH	Setting logging options for RACROUTE REQUEST=DIRAUTH requests. Also, if the DIRAUTH class is active, security label authorization checking is done when a user receives a message sent through the TPUT macro or the TSO SEND, or LISTBC commands. Profiles are not allowed in this class.

Table 39. Resource Classes for z/OS and OS/390 Systems (continued)

Class Name	Description
DLFCLASS	The data lookaside facility.
FACILITY	Miscellaneous uses. Profiles are defined in this class so that resource managers (typically program products or components of MVS or VM) can check a user's access to the profiles when the users take some action. Examples are catalog operations (DFP) and use of the vector facility (an MVS component). RACF does not document all of the resources used in the FACILITY class by other products. For information on the FACILITY class resources used by a specific product (other than RACF itself), see the product's documentation.
FIELD	Fields in RACF profiles (field-level access checking).
GCSFKEYS	Resource group class for CSFKEYS class. [1]
GDASDVOL	Resource group class for DASDVOL class. [1]
GLOBAL	Global access checking table entry. [1]
GMBR	Member class for GLOBAL class (not for use on RACF commands).
GSDSF	Resource group class for SDSF class. [1]
GTERMINL	Resource group class for TERMINAL class. [1]
IBMOPC	Controlling access to OPC/ESA subsystems.
JESINPUT	Conditional access support for commands or jobs entered into the system through a JES input device.
JESJOBS	Controlling the submission and cancellation of jobs by job name.
JESSPOOL	Controlling access to job data sets on the JES spool (that is, SYSIN and SYSOUT data sets).
LOGSTRM	Reserved for MVS/ESA.
NODES	Controlling the following on MVS systems: • Whether jobs are allowed to enter the system from other nodes • Whether jobs that enter the system from other nodes have to pass user identification and password verification checks
NODMBR	Member class for NODES class (not for use on RACF commands).
OPERCMDS	Controlling who can issue operator commands (for example, JES and MVS, and operator commands). [2]
PMBR	Member class for PROGRAM class (not for use on RACF commands).
PROGRAM	Controlled programs (load modules). [1]
PROPCNTL	Controlling if user ID propagation can occur, and if so, for which user IDs (such as the CICS or IMS main task user ID), user ID propagation is *not* to occur.
PSFMPL	Used by PSF to perform security functions for printing, such as separator page labeling, data page labeling, and enforcement of the user printable area.
PTKTDATA	PassTicket key class enables the security administrator to associate a RACF secured signon secret key with a particular mainframe application that uses RACF for user authentication. Examples of such applications are IMS, CICS, TSO, VM, APPC, and MVS batch.
RACGLIST	Class of profiles that hold the results of RACROUTE REQUEST=LIST,GLOBAL=YES or a SETROPTS RACLIST operation.

Table 39. Resource Classes for z/OS and OS/390 Systems (continued)

Class Name	Description
RACFVARS	RACF variables. In this class, profile names, which start with & (ampersand), act as RACF variables that can be specified in profile names in other RACF general resource classes.
RRSFDATA	Used to control RACF remote sharing facility functions.
RVARSMBR	Member class for RACFVARS (not for use on RACF commands).
SCDMBR	Member class for SECDATA class (not for use on RACF commands).
SDSF	Controls the use of authorized commands in the System Display and Search Facility (SDSF). See also GSDSF class.
SECDATA	Security classification of users and data (security levels and security categories). [1]
SECLABEL	If security labels are used, and, if so, their definitions. [2]
SERVAUTH	Contains profiles that are used by servers to check a client's authorization to use the server or to use resources managed by the server.
SERVER	Controlling the server's ability to register with the daemon.
SMESSAGE	Controlling to which users a user can send messages (TSO only).
SOMDOBJS	Controlling the client's ability to invoke the method in the class.
STARTED	Used in preference to the started procedures table to assign an identity during the processing of an MVS START command.
SURROGAT	If surrogate submission is allowed, and if allowed, which user IDs can act as surrogates.
SYSMVIEW	Controlling access by the SystemView for MVS Launch Window to SystemView for MVS applications.
TAPEVOL	Tape volumes.
TEMPDSN	Controlling who can access residual temporary data sets. You cannot create profiles in this resource class.
TERMINAL	Terminals (TSO or VM). See also GTERMINL class.
VTAMAPPL	Controlling who can open ACBs from non-APF authorized programs.
WRITER	Controlling the use of JES writers.
	CICS classes
ACICSPCT	CICS program control table. [2]
BCICSPCT	Resource group class for ACICSPCT class. [1]
CCICSCMD	Used by CICS/ESA 3.1, or later, to verify that a user is permitted to use CICS system programmer commands such as INQUIRE, SET, PERFORM, and COLLECT. [1]
CPSMOBJ	Used by CICSPlex System Manager, which provides a central point of control when running multiple CICS systems, to determine operational controls within a CICS complex.
CPSMXMP	Used by CICSPlex System Manager to identify exemptions from security controls within a CICS complex.
DCICSDCT	CICS destination control table. [2]
ECICSDCT	Resource group class for DCICSDCT class. [1]
FCICSFCT	CICS file control table. [2]
GCICSTRN	Resource group class for TCICSTRN class. [2]

Table 39. Resource Classes for z/OS and OS/390 Systems (continued)

Class Name	Description
GCPSMOBJ	Resource grouping class for CPSMOBJ.
HCICSFCT	Resource group class for FCICSFCT class. [1]
JCICSJCT	CICS journal control table. [2]
KCICSJCT	Resource group class for JCICSJCT class. [1]
MCICSPPT	CICS processing program table. [2]
NCICSPPT	Resource group class for MCICSPPT class. [1]
PCICSPSB	CICS program specification blocks or PSBs
QCICSPSB	Resource group class for PCICSPSB class. [1]
SCICSTST	CICS temporary storage table. [2]
TCICSTRN	CICS transactions.
UCICSTST	Resource group class for SCICSTST class. [1]
VCICSCMD	Resource group class for the CCICSCMD class. [1]
	DB2 classes
DSNADM	DB2 administrative authority class
DSNR	Controls access to DB2 subsystems
GDSNBP	Grouping class for DB2 buffer pool privileges
GDSNCL	Grouping class for DB2 collection privileges
GDSNDB	Grouping class for DB2 database privileges
GDSNJR	Grouping class for Java archive files (JARs)
GDSNPK	Grouping class for DB2 package privileges
GDSNPN	Grouping class for DB2 plan privileges
GDSNSC	Grouping class for DB2 schemas privileges
GDSNSG	Grouping class for DB2 storage group privileges
GDSNSM	Grouping class for DB2 system privileges
GDSNSP	Grouping class for DB2 stored procedure privileges
GDSNTB	Grouping class for DB2 table, index, or view privileges
GDSNTS	Grouping class for DB2 tablespace privileges
GDSNUF	Grouping class for DB2 user-defined function privileges
GDSNUT	Grouping class for DB2 user-defined distinct type privileges
MDSNBP	Member class for DB2 buffer pool privileges
MDSNCL	Member class for DB2 collection privileges
MDSNDB	Member class for DB2 database privileges
MDSNJR	Member class for Java archive files (JARs)
MDSNPK	Member class for DB2 package privileges
MDSNPN	Member class for DB2 plan privileges
MDSNSC	Member class for DB2 schema privileges
MDSNSG	Member class for DB2 storage group privileges
MDSNSM	Member class for DB2 system privileges
MDSNSP	Member class for DB2 stored procedure privileges
MDSNTB	Member class for DB2 table, index, or view privileges

Table 39. Resource Classes for z/OS and OS/390 Systems (continued)

Class Name	Description
MDSNTS	Member class for DB2 tablespace privileges
MDSNUF	Member class for DB2 user-defined function privileges
MDSNUT	Member class for DB2 user-defined distinct type privileges
	DCE classes
DCEUUIDS	Used to define the mapping between a user's RACF user ID and the corresponding DCE principal UUID.
KEYSMSTR	Holds a key to encrypt the DCE password.
	Enterprise Java Beans classes
EJBROLE	Member class for Enterprise Java Beans authorization roles.
GEJBROLE	Grouping class for Enterprise Java Beans authorization roles.
JAVA	Contains profiles that are used by Java for z/OS applications to perform authorization checking for Java for z/OS resources.
	IMS classes
AIMS	Application group names (AGN).
CIMS	Command.
DIMS	Grouping class for command.
FIMS	Field (in data segment).
GIMS	Grouping class for transaction.
HIMS	Grouping class for field.
OIMS	Other.
PIMS	Database.
QIMS	Grouping class for database.
SIMS	Segment (in database).
TIMS	Transaction (trancode).
UIMS	Grouping class for segment.
WIMS	Grouping class for other.
	Information/Management (Tivoli Service Desk) classes
GINFOMAN	Grouping class for Information/Management (Tivoli Service Desk) resources.
INFOMAN	Member class for Information/Management (Tivoli Service Desk) resources.
	LFS/ESA classes
LFSCLASS	Controls access to file services provided by LFS/ESA.
	License Manager class
ILMADMIN	Controls access to the administrative functions of IBM License Manager.
	Lotus Notes for z/OS and Novell Directory Services for OS/390 classes
NDSLINK	Mapping class for Novell Directory Services for OS/390 user identities.
NOTELINK	Mapping class for Lotus Notes for z/OS user identities.
	MQSeries classes
GMQADMIN	Grouping class for MQSeries administrative options. [1]
GMQCHAN	Reserved for MQSeries.

Table 39. Resource Classes for z/OS and OS/390 Systems (continued)

Class Name	Description
GMQNLIST	Grouping class for MQSeries namelists. [1]
GMQPROC	Grouping class for MQSeries processes. [1]
GMQQUEUE	Grouping class for MQSeries queues. [1]
MQADMIN	Protects MQSeries administrative options.
MQCHAN	Reserved for MQSeries.
MQCMDS	Protects MQSeries commands.
MQCONN	Protects MQSeries connections.
MQNLIST	Protects MQSeries namelists.
MQPROC	Protects MQSeries processes.
MQQUEUE	Protects MQSeries queues.
	NetView classes
NETCMDS	Controlling which NetView commands the NetView operator can issue.
NETSPAN	Controlling which NetView commands the NetView operator can issue against the resources in this span.
NVASAPDT	NetView/Access Services.
PTKTVAL	Used by NetView/Access Services Secured Single Signon to store information needed when generating a PassTicket.
RMTOPS	NetView Remote Operations.
RODMMGR	NetView Resource Object Data Manager (RODM).
	Network Authentication Service classes
KERBLINK	Mapping class for user identities of local and foreign principals.
REALM	Used to define the local and foreign realms.
	SMS (DFSMSdfp) classes
MGMTCLAS	SMS management classes.
STORCLAS	SMS storage classes.
SUBSYSNM	Authorizes a subsystem (such as a particular instance of CICS) to open a VSAM ACB and use VSAM Record Level Sharing (RLS) functions.
	Tivoli classes
ROLE	Specifies the complete list of resources and associated access levels that are required to perform the particular job function this role represents and defines which RACF groups are associated with this role.
TMEADMIN	Maps the user IDs of Tivoli administrators to RACF user IDs.
	TSO classes
ACCTNUM	TSO account numbers.
PERFGRP	TSO performance groups.
TSOAUTH	TSO user authorities such as OPER and MOUNT.
TSOPROC	TSO logon procedures.
	z/OS UNIX classes
DIRACC	Controls auditing (using SETROPTS LOGOPTIONS) for access checks for read/write access to HFS directories. Profiles are not allowed in this class.

Table 39. Resource Classes for z/OS and OS/390 Systems (continued)

Class Name	Description
DIRSRCH	Controls auditing (using SETROPTS LOGOPTIONS) of HFS directory searches. Profiles are not allowed in this class.
FSOBJ	Controls auditing (using SETROPTS LOGOPTIONS) for all access checks for HFS objects except directory searches. Controls auditing (using SETROPTS AUDIT) of creation and deletion of HFS objects. Profiles are not allowed in this class.
FSSEC	Controls auditing (using SETROPTS LOGOPTIONS) for changes to the security data (FSP) for HFS objects. Profiles are not allowed in this class.
IPCOBJ	Controlling auditing and logging of IPC security checks.
PROCACT	Controls auditing (using SETROPTS LOGOPTIONS) of functions that look at data from, or affect the processing of, z/OS UNIX processes. Profiles are not allowed in this class.
PROCESS	Controls auditing (using SETROPTS LOGOPTIONS) of changes to UIDs and GIDs of z/OS UNIX processes. Controls auditing (using SETROPTS AUDIT) of dubbing and undubbing of z/OS UNIX processes. Profiles are not allowed in this class.
UNIXMAP	Contains profiles that are used to map z/OS UNIX UIDs to RACF user IDs and z/OS UNIX GIDs to RACF group names.
UNIXPRIV	Contains profiles that are used to grant z/OS UNIX privileges.

Notes:

1. You cannot specify this class name on the GENCMD, GENERIC, and GLOBAL/NOGLOBAL operands of the SETROPTS command.
2. You cannot specify this class name on the GLOBAL operand of SETROPTS or, if you do, the GLOBAL checking is not performed.

Supplied Resource Classes for z/VM and VM Systems

Table 40 lists the supplied classes that can be used on z/VM and VM systems. These classes are primarily relevant if you share your RACF database with a z/VM or VM system.

Table 40. Resource Classes for z/VM and VM Systems

Class Name	Description
DIRECTRY	Protection of shared file system (SFS) directories.
FACILITY	Miscellaneous uses. Profiles are defined in this class so resource managers (typically program products or components of MVS or VM) can check a user's access to the profiles when the users take some action. Examples are using combinations of options for tape mounts, and use of the RACROUTE interface. RACF does not document all of the resources used in the FACILITY class by other products. For information on the FACILITY class resources used by a specific product (other than RACF itself), see that product's documentation.
FIELD	Fields in RACF profiles (field-level access checking).
FILE	Protection of shared file system (SFS) files.
GLOBAL	Global access checking. [1]
GMBR	Member class for GLOBAL class (not for use on RACF commands).

Table 40. Resource Classes for z/VM and VM Systems (continued)

Class Name	Description
GTERMINL	Terminals whose IDs do not fit into generic profile naming conventions. [1]
PSFMPL	When class is active, PSF/VM performs separator and data page labeling as well as auditing.
PTKTDATA	PassTicket key class.
PTKTVAL	Used by NetView/Access Services Secured Single Signon to store information needed when generating a PassTicket.
RACFVARS	RACF variables. In this class, profile names, which start with & (ampersand), act as RACF variables that can be specified in profile names in other RACF general resource classes.
RVARSMBR	Member class for RACFVARS (not for use on RACF commands).
SCDMBR	Member class for SECDATA class (not for use on RACF commands).
SECDATA	Security classification of users and data (security levels and security categories). [1]
SECLABEL	If security labels are used and, if so, their definitions. [2]
SFSCMD	Controls the use of shared file system (SFS) administrator and operator commands.
TAPEVOL	Tape volumes.
TERMINAL	Terminals (TSO or VM). See also GTERMINL class.
VMBATCH	Alternate user IDs.
VMBR	Member class for VMEVENT class (not for use on RACF commands).
VMCMD	Certain CP commands and other requests on VM.
VMEVENT	Auditing and controlling security-related events (called VM events) on VM systems.
VMMAC	Used in conjunction with the SECLABEL class to provide security label authorization for some VM events. Profiles are not allowed in this class.
VMMDISK	VM minidisks.
VMNODE	RSCS nodes.
VMRDR	VM unit record devices (virtual reader, virtual printer, and virtual punch).
VMSEGMT	Restricted segments, which can be named saved segments (NSS) and discontiguous saved segments (DCSS).
VXMBR	Member class for VMXEVENT class (not for use on RACF commands).
VMXEVENT	Auditing and controlling security-related events (called VM events) on VM systems.
VMPOSIX	Contains profiles used by OpenExtensions VM.
WRITER	VM print devices.

Notes:

1. You cannot specify this class name on the GENCMD, GENERIC, and GLOBAL/NOGLOBAL operands of the SETROPTS command.
2. You cannot specify this class name on the GLOBAL operand of the SETROPTS command or, if you do, the GLOBAL checking is not performed.

Appendix B. Summary of RACF Commands and Authorities

This appendix summarizes the RACF commands and authorities.

Summary of Commands and Their Functions

RACF commands allow you to list, modify, add, and delete profiles for users, groups, connect entries, and resources. Table 41 shows, in alphabetic order, each of the commands and its functions.

Table 41. Functions of RACF Commands

RACF Command	Command Functions
ADDGROUP	• Define one or more new groups as a subgroup of an existing group. • Specify a model data set profile for a group. • Define default DFP information for a group. • Define the z/OS UNIX information for a group. • Define a group as a universal group.
ADDSD	• RACF-protect one or more existing data sets. • RACF-define one or more data sets brought from another system where they were RACF-protected. • RACF-define generic data set profiles. • Create a new data set model profile.
ADDUSER	• Define one or more new users and connect the users to their default connect group. • Specify a model data set profile for a user. • Define CICS operator information. • Define default DFP information for a user. • Define the preferred national language. • Define default operator information. • Define default TSO logon information for a user. • Define default work attributes. • Define the z/OS UNIX information for a user. • Define the DCE information for a user. • Define default NetView operator information. • Define the COMMAND field of the logon panel. • Define the LNOTES and NDS information for a user. • Define access checking with the RESTRICTED and NORESTRICTED keywords. • Define the KERB information for a user.
ALTDSD	• Change one or more discrete or generic data set profiles. • Protect a single volume of a multivolume, non-VSAM DASD data set. • Remove protection from a single volume of a multivolume, non-VSAM DASD data set.
ALTGROUP	• Change the information in one or more group profiles (such as the superior group, owner, or model profile name). • Change or delete the default DFP information for a group. • Add, change, or delete the information for the z/OS UNIX group.

Table 41. Functions of RACF Commands (continued)

RACF Command	Command Functions
ALTUSER	• Change the information in one or more user profiles (such as the owner, universal access authority, or security level). • Revoke or reestablish one or more users' privileges to access the system. • Specify logging of information about the user, such as the commands the user issues. • Change or delete CICS operator information. • Change or delete the default DFP information for a user. • Change the preferred national language. • Change or delete the default operator information. • Change or delete the default TSO logon information for a user. • Change or delete the default work attributes. • Add, change, or delete the information for the z/OS UNIX user. • Change the DCE information for a user. • Change or delete NetView operator information. • Manipulate the COMMAND field of the logon panel. • Change the LNOTES and NDS information for a user. • Change access checking methods with the RESTRICTED and NORESTRICTED keywords. • Add or change the KERB information for a user.
CONNECT	• Connect one or more users to a group. • Modify one or more users' connection to a group. • Revoke or reestablish one or more users' privileges to access the system.
DELDSD	• Delete one or more discrete or generic data set profiles. • Delete a discrete data set profile for a tape data set, while retaining the data set name in the TVTOC. • Remove a data set profile, but leave the data set RACF-indicated, when moving a RACF-protected data set to another system that has RACF.
DELGROUP	• Delete one or more groups and their relationship to the superior group.
DELUSER	• Delete one or more users and remove all of their connections to RACF groups.
DISPLAY	• Display users signed on to a RACF subsystem.
HELP	• Display the function and proper syntax of RACF commands.
LISTDSD	• List the details of one or more discrete or generic data set profiles, including the users and groups authorized to access the data sets. • Determine the most specific matching generic profile for a data set. • Perform a local refresh of generic DATASET profiles.
LISTGRP	• List the details of one or more group profiles, including the users connected to the group. • List only the information contained in a specific segment (RACF, DFP, or OMVS) of the group profile. • Display limited information if the group is a UNIVERSAL group.
LISTUSER	• List the details of one or more user profiles, including all of the groups to which each user is connected. • List only the information contained in a specific segment (for example, DFP or OMVS information) of a user profile.
PASSWORD	• Change one or more users' passwords. • Change one or more users' password change interval. • Reset one or more users' passwords to a known default value.
PERMIT	• Give or remove authority to access a resource to specific users or groups. • Change the level of access authority to a resource for specific users or groups. • Copy the list of authorized users from one resource profile to another. • Delete an existing access list.

Table 41. Functions of RACF Commands (continued)

RACF Command	Command Functions
RACDCERT	• List information about the existing certificate definitions for a specified RACF-defined user ID or the requester's user ID. • Add a certificate definition and associate it with a specified RACF-defined user ID or the requester's user ID and set the TRUST flag. • Check to see if a certificate has been defined to RACF. • Alter the TRUST flag for an existing definition. • Delete an existing definition. • Add or remove a certificate to a key ring. • Create, delete, or list an existing key ring. • Generate a public or private key pair and certificate. • Write a certificate to a data set. • Create a certificate request. • Create, alter, delete, or list a certificate name filter. • Gather diagnostic information.
RACLINK	• Define, approve, and delete (undefine) a user ID association. • List information related to a user ID association. • Establish password synchronization between user IDs.
RALTER	• Change the discrete or generic profiles for one or more resources whose class is defined in the class descriptor table. • Maintain the global access checking tables. • Maintain security category and security level tables. • List the encryption keys used if the profile has a KERB segment. • Maintain DLFDATA, SESSION, SSIGNON, and STDATA segment information in the profiles. • Change profiles associated with a SystemView for MVS application.
RDEFINE	• RACF-protect by a discrete or generic profile one or more resources whose class is defined in the class descriptor table. • Define the global access checking tables. • Define security category and security level tables. • Define the encryption keys used if the profile has a KERB segment. • Define DLFDATA, SESSION, SSIGNON, and STDATA segment information in the profiles. • Define the list of classes for which RACF is to save RACLISTed results on the RACF database. • Define profiles associated with a SystemView for MVS application. • Create, alter, or delete additional criteria for a certificate name filter.
RDELETE	• Remove RACF-protection from one or more resources whose class is defined in the class descriptor table. • Delete the global access checking tables. • Delete the security category and security level tables. • Delete a class from the list of classes for which RACF saves RACLISTed results on the RACF database.
REMOVE	• Remove one or more users from a group and assign a new owner for any group data sets owned by the users.
RESTART	• Restart a function in the RACF subsystem address space. • Restart the connection to a specific member system on a multisystem node.
RLIST	• List the details of discrete or generic profiles for one or more resources whose class is defined in the class descriptor table. • List the contents of the DLFDATA, SESSION, SSIGNON, and STDATA segments in the profiles. • List the encryption keys used if the profile has a KERB segment. • Perform a local refresh of generic general resource profiles.
RVARY	• Dynamically deactivate and reactivate the RACF function. • Dynamically deactivate and reactivate the RACF primary and backup database. • Switch the primary and backup RACF databases. • Deactivate resource protection, for any resource whose class is defined in the class descriptor table, while RACF is deactivated. • Select operational mode when RACF is enabled for sysplex communication.
SEARCH	• List the RACF profile names that meet a search criteria for a class of resources. • Create a CLIST of the RACF profile names that meet a search criteria for a class of resources.

Table 41. Functions of RACF Commands (continued)

RACF Command	Command Functions
SET	• List information related to RACF remote sharing facility (RRSF) on the local node. • Specify the name of a member of the RACF parameter library to be processed by RRSF. • Set tracing on or off for specified operands. • Specify options for automatic command direction.
SETROPTS	Dynamically set system-wide options relating to resource protection, specifically: • Choose the resource classes that RACF is to protect. • Gather and display RACF statistics. • Set the universal access authority (UACC) for terminals. • Set the KERBLVL to be used if a profile has a KERB segment. • Specify logging of certain RACF commands and events. • Permit list-of-groups access checking. • Display options currently in effect. • Enable or disable generic profile checking on either a class-by-class or system-wide level. • Control user password syntax rules. • Establish password syntax rules. • Activate password processing for checking previous passwords, limit invalid password attempts, and warn of password expiration. • Control global access checking for selected individual resources or generic names with selected generalized access rules. • Set the passwords for authorizing use of the RVARY command. • Initiate refreshing of in-storage generic profile lists and global access checking tables. • Enable or disable shared generic profiles for general resources in common storage. • Enable or disable shared profiles through RACLIST processing for general resources. • Activate or deactivate auditing of access attempts to RACF-protected resources based on installation-defined security levels. • Activate enhanced generic naming. • Control the use of automatic data set protection (ADSP). • Activate profile modeling for GDG, group, and user data sets. • Activate protection for data sets with single-level names. • Control logging of real data set names. • Control the job entry subsystem (JES) options. • Activate tape data set protection. • Control whether or not data sets must be RACF-protected. • Control the erasure of scratched DASD data sets. • Activate program control. • Control whether a profile creator's user ID is automatically added to the profile's access list.
SIGNOFF	• Sign off users from a RACF subsystem.
STOP	• Stop the RACF subsystem address space.
TARGET	• List the controls and operational characteristics of the specified target RRSF nodes. • Specify the name of the target RRSF node. • Request an operational state for connection to the target RRSF node. • Delete an RRSF node from the local node. • Specify a description of the target RRSF node. • Purge the workspace data sets managed by RRSF in the RACF subsystem address space. • Specify the protocol type for the transport mechanism to be used in communication between the two RRSF nodes. • Specify the relationship between the target RRSF node and the node being configured. • Specify a prefix for the workspace data sets allocated by and used by RRSF for each target node. • Specify the characteristics of the workspace data sets associated with the node being defined to RRSF. • Specify the name of a multisystem node. • Identify the main system in a multisystem RRSF node.

Summary of Authorities and Commands

This section summarizes the attributes and authorities that can be assigned to users, and the RACF commands and operands that can be issued for each authority. It provides information for the following categories:

- User attributes (SPECIAL or group-SPECIAL, AUDITOR or group-AUDITOR, OPERATIONS or group-OPERATIONS, and CLAUTH)
- Group authorities (USE, CREATE, CONNECT, JOIN)
- Access authorities (NONE, EXECUTE, READ, UPDATE, CONTROL, and ALTER)
- Other authorities not specified above.

The SPECIAL or group-SPECIAL Attribute

If you have the SPECIAL or group-SPECIAL attribute, you can issue the commands and operands shown in Table 42.

Table 42. Commands and Operands You Can Issue If You Have the SPECIAL or group-SPECIAL Attribute

ADDSD[1]	with all operands
ADDGROUP	with all operands
ADDUSER	with all operands, but for group-SPECIAL user only when user also has CLAUTH(USER)
ALTDSD[1]	with all operands except GLOBALAUDIT
ALTGROUP	with all operands
ALTUSER	with all operands except UAUDIT or NOUAUDIT. Also, you must have the SPECIAL attribute to use the NOEXPIRED operand or to issue the NOCLAUTH operand for a class name that is not in the class descriptor table (group-SPECIAL does not suffice).
CONNECT	with all operands
DELDSD[1]	with all operands
DELGROUP	with all operands
DELUSER	with all operands
LISTDSD[1]	with all operands
LISTGRP	with all operands
LISTUSER	with all operands
PASSWORD	with all operands
PERMIT	with all operands
RALTER	with all operands except GLOBALAUDIT
RACDCERT	with all operands. You must have the SPECIAL attribute to issue the RACDCERT command (group-SPECIAL does not suffice).
RACLINK	with all operands
RDEFINE	with all operands
RDELETE	with all operands
REMOVE	with all operands
RLIST	with all operands
SEARCH	with all operands
SETROPTS	with all operands except AUDIT, NOAUDIT, CMDVIOL, NOCMDVIOL, APPLAUDIT, NOAPPLAUDIT, LOGOPTIONS, OPERAUDIT, NOOPERAUDIT, SAUDIT, NOSAUDIT, SECLABELAUDIT, NOSECLABELAUDIT, SECLEVELAUDIT, and NOSECLEVELAUDIT, which require the AUDITOR attribute. Users with the group-SPECIAL attribute can only issue REFRESH GENERIC and LIST.

[1] This command applies to z/OS and OS/390 systems. However, you can issue this command on a VM system to maintain a RACF database that is shared among z/OS, OS/390, and VM systems.

The AUDITOR or group-AUDITOR Attribute

If you have the AUDITOR or group-AUDITOR attribute, you can issue the commands and operands shown in Table 43.

Table 43. Commands and Operands You Can Issue If You Have the AUDITOR or group-AUDITOR Attribute

ALTDSD[1]	only with GLOBALAUDIT
ALTUSER	only with UAUDIT or NOUAUDIT
LISTDSD[1]	with all operands, lists GLOBALAUDIT option
LISTGRP	with all operands
LISTUSER	with all operands, lists UAUDIT or NOUAUDIT operand
RALTER	only with GLOBALAUDIT
RLIST	with all operands, lists GLOBALAUDIT option
SEARCH	with all operands
SETROPTS	only with AUDIT, NOAUDIT, CMDVIOL or NOCMDVIOL, LOGOPTIONS, OPERAUDIT, NOOPERAUDIT, SAUDIT, NOSAUDIT, SECLABELAUDIT, NOSECLABELAUDIT, SECLEVELAUDIT, NOSECLEVELAUDIT, LIST, or REFRESH GENERIC

[1] This command applies to z/OS and OS/390 systems. However, you can issue this command on a VM system to maintain a RACF database that is shared among z/OS, OS/390, and VM systems.

The OPERATIONS or group-OPERATIONS Attribute

If you have the OPERATIONS or group-OPERATIONS attribute, you can issue the commands and operands shown in Table 44.

Table 44. Commands and Operands You Can Issue If You Have the OPERATIONS or group-OPERATIONS Attribute

ADDSD[1]	when adding new profiles for group data sets
LISTDSD[1]	with all operands except GLOBALAUDIT
RLIST	with all operands except GLOBALAUDIT
SEARCH	with all operands
SETROPTS	only with REFRESH

[1] This command applies to z/OS and OS/390 systems. However, you can issue this command on a VM system to maintain a RACF database that is shared among z/OS, OS/390, and VM systems.

The CLAUTH Attribute

If you have the CLAUTH attribute, you can issue the commands and operands shown in Table 45.

Table 45. Commands and Operands You Can Issue If You Have the CLAUTH Attribute

ADDUSER[1]	with all operands except OPERATIONS, NOOPERATIONS, SPECIAL, NOSPECIAL, AUDITOR or NOAUDITOR
ALTUSER[2]	only with CLAUTH or NOCLAUTH
RALTER[3], [5]	only with ADDVOL
RDEFINE[4], [5]	with all operands (special rules apply to ADDMEM)
SETROPTS[4]	only with REFRESH GLOBAL or REFRESH GENERIC

Table 45. Commands and Operands You Can Issue If You Have the CLAUTH Attribute (continued)

[1] This command applies when you have the CLAUTH attribute of USER and you either are the owner of a group, have JOIN authority in the default group specified in the command, or the profile is within the scope of a group in which you have the group-SPECIAL attribute.
[2] This command applies when you have the CLAUTH attribute for the class to be added or deleted, the class name is in the class descriptor table (CDT), *and* either you are the owner of the user's profile, or the profile is within the scope of a group in which you have the group-SPECIAL attribute.
[3] This command applies when you have the CLAUTH attribute of TAPEVOL and you also have sufficient authority to issue the command.
[4] These commands apply when you have the CLAUTH attribute for the specified class.
[5] For ADDMEM, special rules apply, depending on access to individual resources. For more information, see the description of the ADDMEM operand in *z/OS SecureWay Security Server RACF Command Language Reference*.

Group Authority

If you have a group authority, you can issue the commands and operands shown in Table 46.

Table 46. Commands and Operands You Can Issue If You Have a Group Authority

Group Authorities	Commands and Operands You Can Issue If You Have This Authority	
USE	For group resources, the authority allowed the group.	
CREATE	**ADDSD**[1]	with all operands except NOSET
CONNECT	**ADDSD**[1,5]	with all operands except NOSET
	ALTUSER	only with GROUP, AUTHORITY or UACC
	CONNECT	with all operands except SPECIAL, NOSPECIAL, OPERATIONS, NOOPERATIONS, AUDITOR, or NOAUDITOR
	LISTGRP	only with group name
	REMOVE	with all operands
JOIN	**ADDGROUP**[2]	with all operands
	ADDSD[1,5]	with all operands except NOSET
	ADDUSER[3]	with all operands except OPERATIONS, SPECIAL or AUDITOR
	ALTGROUP[4]	only with SUPGROUP
	ALTUSER	only with GROUP, AUTHORITY or UACC
	CONNECT	with all operands except SPECIAL, NOSPECIAL, OPERATIONS or NOOPERATIONS
	DELGROUP[2]	with all operands
	LISTGRP	only with group name
	REMOVE	with all operands

Table 46. Commands and Operands You Can Issue If You Have a Group Authority (continued)

Group Authorities	Commands and Operands You Can Issue If You Have This Authority
[1] This command applies to group data sets only. [2] This command applies to the superior group. [3] This command applies only if you have the JOIN group authority in the default group specified in the ADDUSER command and if you also have the CLAUTH(USER) attribute. [4] This command applies to current and new superior groups. You can have JOIN authority in one group and be owner of or be connected with the group-SPECIAL attribute to another group. [5] This command applies to z/OS and OS/390 systems. However, you can issue this command on a VM system to maintain a RACF database that is shared among z/OS, OS/390, and VM systems.	

Access Authority

If you have an access authority, you can issue the commands and operands shown in Table 47.

Table 47. Commands and Operands You Can Issue If You Have an Access Authority

Access Authorities	Commands and Operands You Can Issue If You Have This Authority	
NONE EXECUTE	None	
READ UPDATE CONTROL	**LISTDSD**[1]	with all operands except AUTHUSER or ALL
	RLIST	with all operands except AUTHUSER or ALL
	SEARCH	with all operands
ALTER	**ALTDSD**[1,2]	with all operands except OWNER, NOSET or GLOBALAUDIT
	DELDSD[1,2]	with all operands except NOSET
	LISTDSD[1,2]	with all operands
	PERMIT[2]	with all operands
	RALTER[2]	with all operands except OWNER, ADDVOL[3] or GLOBALAUDIT
	RDELETE[2]	with all operands
	RLIST[2]	with all operands
[1] This command applies to z/OS and OS/390 systems. However, you can issue this command on a VM system to maintain a RACF database that is shared among z/OS, OS/390, and VM systems. [2] This command applies to discrete profiles only. [3] This command applies to ADDVOL operand only if you also have CLAUTH attribute for TAPEVOL.		

Profile Ownership Authority

If you own a profile, you can issue the commands and operands shown in Table 48 on page 581.

Table 48. Commands and Operands You Can Issue If You Own a Profile

Owner of RACF Profile	Commands and Operands You Can Issue If You Are the Owner	
Owner of user profile	**ALTUSER**[1]	only with user ID, NAME, OWNER, DFLTGRP, DATA, GRPACC, NOGRPACC, ADSP, NOADSP, REVOKE, RESUME, PASSWORD, NOPASSWORD, OIDCARD, NOOIDCARD, CLAUTH or NOCLAUTH
	DELUSER	with all operands
	LISTUSER	with all operands
	PASSWORD	only with USER
	RACLINK	with all operands
Owner of group profile	**ADDGROUP**[2]	with all operands
	ADDUSER[3]	with all operands except OPERATIONS, SPECIAL or AUDITOR
	ALTGROUP[4]	with all operands
	ALTUSER	only with GROUP, AUTHORITY or UACC
	CONNECT	with all operands except SPECIAL, NOSPECIAL, OPERATIONS or NOOPERATIONS
	DELGROUP[5]	with all operands
	LISTGRP	with all operands
	REMOVE	with all operands
Owner of resource profile	**ALTDSD**[7]	with all operands except NOSET or GLOBALAUDIT
	DELDSD[7]	with all operands except NOSET
	LISTDSD[7]	with all operands
	PERMIT	with all operands
	RALTER[6]	with all operands except GLOBALAUDIT
	RDELETE	with all operands
	RLIST	with all operands
	SEARCH	with all operands

[1] This command applies to CLAUTH or NOCLAUTH only if you have the CLAUTH attribute for the class to be added or deleted, and the class name is in the class descriptor table (CDT).

[2] This command applies to the superior group.

[3] This command applies to the default group specified and only if you have the CLAUTH attribute of USER.

[4] This command applies to current and new superior groups. You can have JOIN authority in one group and be owner of another group.

[5] This command applies to the superior group or group to be deleted.

[6] This command applies to the ADDVOL operand only when you also have CLAUTH attribute of TAPEVOL.

[7] This command applies to z/OS and OS/390 systems. However, you can issue this command on a VM system to maintain a RACF database that is shared among z/OS, OS/390, and VM systems.

Other Authorities

Table 49 on page 582 shows the commands and operands you can issue for reasons not already covered previously.

Commands and Authorities

Table 49. Commands and Operands You Can Issue for Miscellaneous Reasons

User ID Relationship	Commands and Operands You Can Issue for Miscellaneous Reasons	
User ID is current user	**ALTUSER**	only with NAME or DFLTGRP
	LISTUSER	only with user ID
	PASSWORD	only with PASSWORD or INTERVAL
User ID is high-level qualifier of data set name (or qualifier supplied by a command installation exit)	**ADDSD**	with all operands
	ALTDSD	with all operands except OWNER or GLOBALAUDIT
	DELDSD	with all operands
	LISTDSD	with all operands
	PERMIT	with all operands
	SEARCH	with all operands
None	**RVARY**[1]	with all operands
None	RACF MVS Operator Commands:	
	DISPLAY	Authority granted by OPERCMDS class: See Table 23 on page 278 and *z/OS SecureWay Security Server RACF Command Language Reference*.
	RESTART	Authority granted by OPERCMDS class
	SET	Authority granted by OPERCMDS class
	SIGNOFF	Authority granted by OPERCMDS class: See Table 23 on page 278 and *z/OS SecureWay Security Server RACF Command Language Reference*.
	STOP	Authority granted by OPERCMDS class
	TARGET	Authority granted by OPERCMDS class
	Any RACF TSO command issued as an operator command	

[1] Although no special authority is needed to issue this command, the system operator must supply the appropriate RVARY password, as established by the SETROPTS command with the RVARYPW operand, to approve any change in RACF status.

Appendix C. RACF Profile Summaries

This appendix summarizes the contents of each type of RACF profile.

User Profile Contents Summary

Table 50. User Profile Contents: ADDUSER Command

Operand of ADDUSER Command:	**Description**
userid	The user profile name (required)
NAME('*user-name*')	The user's name
DATA('*installation-defined-data*')	Installation-defined data
DFLTGRP(*groupname*)	The user's default group
OWNER(*userid* or *groupname*)	The owner of the profile
UACC(*access-authority*)	The default universal access authority for all new profiles the user defines while connected to the default group
AUTHORITY(*group-authority*)	The user's level of group authority (USE, CREATE, CONNECT, or JOIN)
CLAUTH | NOCLAUTH	The classes for which the user is authorized to define profiles (or none)
AUDITOR | NOAUDITOR	The user's AUDITOR attribute setting
OPERATIONS | NOOPERATIONS	The user's OPERATIONS attribute setting
SPECIAL | NOSPECIAL	The user's SPECIAL attribute setting
OIDCARD | NOOIDCARD	A setting that indicates whether the user must supply an operator identification card during logon
PASSWORD | NOPASSWORD	The user's initial password (or none). For a user with the NOOIDCARD attribute, NOPASSWORD indicates that the user has the PROTECTED attribute.
WHEN([DAYS(*day-info*)][TIME(*time-info*)])	The times when the user is authorized to access the system from a terminal
ADDCATEGORY(*category-name* ...)	The security categories that the user is allowed to access
SECLABEL(*security-label*)	The user's default security label
SECLEVEL(*security-level*)	The user's default security level
ADSP | NOADSP	The setting for automatic data set protection for data sets created by this user
GRPACC | NOGRPACC	The setting for group access to any group data sets protected data set profiles defined by the user
MODEL(*dsname*)	The name of a profile to be used as a model
RESTRICTED | NORESTRICTED	Indicates whether global access checking, the `ID(*)` entry on the access list, or the UACC is used to allow this user to access a resource
CICS	CICS segment information:
OPCLASS(*operator-class1*,*operator-class2*,...)	The CICS operator classes for which the user will receive BMS (basic mapping support) messages
OPIDENT(*operator-id*)	The user's CICS operator identifier

Table 50. User Profile Contents: ADDUSER Command (continued)

Operand of ADDUSER Command:	Description
OPPRTY(*operator-priority*)	The user's CICS operator priority
TIMEOUT(*timeout-value*)	The amount of time that the user can be idle before being signed off by CICS
XRFSOFF(FORCE | NOFORCE)	A setting that indicates whether the user will be signed off by CICS when an XRF takeover occurs
DCE	DCE segment information:
AUTOLOGIN(NO|YES)	Indicates whether DCE is to automatically log this user in to DCE
DCENAME(*user-principal-name*)	The DCE principal name
HOMECELL(*dce-cell-name*)	The DCE cell name for this RACF user
HOMEUUID(*home-cell-uuid*)	The DCE UUID for the cell that this user is defined to
UUID(*universal-unique-identifier*)	The DCE UUID of the DCE principal defined in DCENAME
DFP	DFP segment information:
DATAAPPL(*application-name*)	The DFP data application identifier
DATACLAS(*data-class-name*)	The default DFP data class
MGMTCLAS(*management-class-name*)	The default DFP management class
STORCLAS(*storage-class-name*)	The default DFP storage class
KERB	KERB segment information:
KERBNAME(*local_principal_name*)	The user's local principal name
ENCRYPT([DES | NODES] [DES3 | NODES3] [DESD | NODESD])	The user's key encryption options.
MAXTKTLFE(*max_ticket_life*)	The user's maximum ticket life
LANGUAGE	LANGUAGE segment information:
PRIMARY(*language*)	The user's primary national language
SECONDARY(*language*)	The user's secondary national language
LNOTES	LNOTES segment information:
SNAME(*short-name*)	The user's Lotus Notes for z/OS short name
NDS	NDS segment information:
UNAME(*user-name*)	The user's Novell Directory Services for OS/390 user name
NETVIEW	NETVIEW segment information:
CONSNAME(*console-name*)	MCS console identifier
CTL(GENERAL | GLOBAL | SPECIFIC)	Specifies GLOBAL, GENERAL, or SPECIFIC control
DOMAINS(*domain-names*)	Domain identifier
IC('*commands*')	Initial command or list of commands to be executed by NetView when this NetView operator logs on
MSGRECVR(NO | YES)	Indicates whether the operator will receive unsolicited messages
NGMFADMN(NO | YES)	Indicates whether this operator can use the NetView graphic monitor facility
OPCLASS(*classes*)	Class of the operator
OMVS	OMVS segment information:
HOME(*initial-directory-name*)	The user's z/OS UNIX home directory path name

Table 50. User Profile Contents: ADDUSER Command (continued)

Operand of ADDUSER Command:	Description
PROGRAM(*program-name*)	The user's initial z/OS UNIX shell program
UID(*user-identifier*)	The user's z/OS UNIX user identifier
CPUTIMEMAX(*cpu-time*)	User's z/OS UNIX `RLIMIT_CPU` (maximum CPU time)
ASSIZEMAX(*address-space-size*)	User's z/OS UNIX `RLIMIT_AS` (maximum address space size)
FILEPROCMAX(*files-per-process*)	User's z/OS UNIX maximum number of files per process
PROCUSERMAX(*processes-per-UID*)	User's z/OS UNIX maximum number of processes per UID
THREADSMAX(*threads-per-process*)	User's z/OS UNIX maximum number of threads per process
MMAPAREAMAX(*memory-map-size*)	User's z/OS UNIX maximum memory map size
OPERPARM	OPERPARM segment information:
ALTGRP(*alternate-console-group*)	The alternate console group to be used for recovery of the user's MCS console
AUTH(MASTER | ALL | INFO | any others)	The authority that the user's MCS console has to issue operator commands
AUTO(YES | NO)	A setting that indicates whether the user's MCS console can receive messages that have been automated by the Message Processing Facility (MPF) in the sysplex
CMDSYS(*system-name* | *)	The system to which commands issued from the user's MCS console are to be sent
DOM(NORMAL | ALL | NONE)	A setting that indicates whether the user's MCS console receives delete operator message (DOM) requests
KEY(*searching-key*)	The console key to be associated with the user's MCS console
LEVEL(*message-level*)	The types of messages that the user's MCS console is to receive
LOGCMDRESP(SYSTEM | NO)	A setting that indicates whether command responses from the user's MCS console are to be logged
MFORM(*message-format*)	The format in which messages are to be displayed at the user's MCS console
MIGID(YES | NO)	The migration ID that is to be assigned to the user's MCS console
MONITOR(*events*)	The information that should be displayed at the user's MCS console when jobs, TSO sessions, or data set status are being monitored
MSCOPE(*system-names* | * | *ALL)	The systems from which the user's MCS console can receive messages that are not directed to a specific console
ROUTCODE(ALL | NONE | *routing-codes*)	The routing codes of messages that the user's MCS console is to receive
STORAGE(*amount*)	The amount of storage in the TSO user's address space that can be used for message queueing to the user's MCS console
UD(YES | NO)	A setting that indicates whether the user's MCS console is to receive undelivered messages
OVM	OVM segment information:
FSROOT(*directory-name*)	The user's OpenExtensions VM file system directory name

Table 50. User Profile Contents: ADDUSER Command (continued)

Operand of ADDUSER Command:	Description
HOME(*initial-directory-name*)	The user's OpenExtensions VM home directory path name
PROGRAM(*program-name*)	The user's initial OpenExtensions VM shell program
UID(*user-identifier*)	The user's OpenExtensions VM user identifier
TSO	TSO segment information:
ACCTNUM(*account-number*)	The user's default TSO account number at logon
DEST(*destination-id*)	The default destination to which the user can route dynamically allocated SYSOUT data sets
HOLDCLASS(*hold-class*)	The user's default hold class
JOBCLASS(*job-class*)	The user's default job class
MAXSIZE(*maximum-region-size*)	The maximum region size the user can request during logon
MSGCLASS(*message-class*)	The user's default message class
PROC(*logon-procedure-name*)	The name of the user's default logon procedure
SECLABEL(*security-label*)	The security label specified by the user during logon to TSO
SIZE(*default-region-size*)	The minimum region size if the user does not request a region size during logon
SYSOUTCLASS(*sysout-class*)	The user's default SYSOUT class
COMMAND	The default command at logon
UNIT(*unit-name*)	The default name of a device or group of devices that a procedure uses for allocations
USERDATA(*user-data*)	Installation-defined data
WORKATTR	WORKATTR segment information:
WAACCNT(*account-number*)	An account number for APPC/MVS processing
WAADDR*n* (*address-line*)	Up to four additional address lines for SYSOUT delivery
WABLDG(*building*)	The building to which SYSOUT information is to be delivered
WADEPT(*department*)	The department to which SYSOUT information is to be delivered
WANAME(*name*)	The name of the user to which SYSOUT information is to be delivered
WAROOM(*room*)	The room to which SYSOUT information is to be delivered

Table 51. User Profile Contents: RACDCERT Command

Operand of RACDCERT Command:	Description
ADD *dataset-name*	Assigns a digital certificate for the user ID
TRUST | NOTRUST	Specifies the trust status of the certificate
WITHLABEL('*label-name*')	Specifies the label associated with the certificate
PASSWORD('*pkcs12-password*')	Specifies the password associated with the PKCS #12 certificate package
ICSF	Specifies that RACF should attempt to store the private key associated with this certificate in ICSF
MAP *dataset-name*	Associates a certificate name filter with a user ID
WITHLABEL	Specifies the label of the certificate name filter
IDNFILTER	Creates the name of the DIGTNMAP class profile that contains the certificate name filter associated with this user ID
SDNFILTER	
GENCERT(*request-dataset-name*)	Creates a public/private key pair and generates a digital certificate

Table 51. User Profile Contents: RACDCERT Command (continued)

Operand of RACDCERT Command:	Description
SUBJECTSDN	Specifies the subject's X.509 distinguished name

Table 52. User Profile Contents: RACLINK Command

Operand of RACLINK Command:	Description
PEER(PWSYNC)	A peer-to-peer association between two user IDs that provides password synchronization and two-way command direction. The association can be ended by either user.
PEER(NOPWSYNC)	A peer-to-peer association between two user IDs that provides two-way command direction, but not password synchronization. The association can be ended by either user.
MANAGED	An association between two user IDs that provides one-way command direction, but not password synchronization. The association can be ended by either user.

Group Profile Contents Summary

Table 53. Group Profile Contents: ADDGROUP Command

Operand of ADDGROUP Command:	Description
groupname	The group profile name (required)
OWNER(*userid or groupname*)	The owner of the profile
SUPGROUP(*groupname*)	The profile's superior group
DATA('*installation-defined-data*')	Installation-defined data
MODEL(*dsname*)	The name of a profile to be used as a model
TERMUACC | NOTERMUACC	The type of terminal authorization for the group
UNIVERSAL	Universal group
DFP	DFP segment information:
DATAAPPL(*application-name*)	The DFP data application identifier
DATACLAS(*data-class-name*)	The default DFP data class
MGMTCLAS(*management-class-name*)	The default DFP management class
STORCLAS(*storage-class-name*)	The default DFP storage class
OMVS	OMVS segment information:
GID(*group-identifier*)	The group's z/OS UNIX group identifier
OVM	OVM segment information:
GID(*group-identifier*)	The group's OpenExtensions VM group identifier
TME	TME segment information:
ROLES(*profile-name ...*)	Specifies the complete list of roles that refer to this group

Connect Profile Contents Summary

Table 54. Connect Profile Contents: CONNECT Command

Operand of CONNECT Command:	Description
userid...	The user or users to be connected
GROUP(*groupname*)	The group to be connected to
OWNER(*userid* or *groupname*)	The owner of the connect profile

Profiles

Table 54. Connect Profile Contents: CONNECT Command (continued)

Operand of CONNECT Command:	Description
AUTHORITY(*group-authority*)	The user's level of group authority (USE, CREATE, CONNECT, or JOIN)
UACC[(*access-authority*)]	The default universal access authority for all new resources the user defines while connected to the group
AUDITOR | NOAUDITOR	The user's group-AUDITOR attribute setting while the user is connected to the group
OPERATIONS | NOOPERATIONS	The user's group-OPERATIONS attribute setting while the user is connected to the group
SPECIAL | NOSPECIAL	The user's group-SPECIAL attribute setting while the user is connected to the group
REVOKE[(*date*)]	A setting that indicates that RACF is to prevent the user from logging on and connecting to the group, either immediately or as of the specified date
RESUME[(*date*)]	A setting that indicates that RACF is to allow the user to log on and connect to the group, either immediately or as of the specified date
ADSP | NOADSP	The setting for automatic data set protection for data sets created by this user while the user is connected to the group
GRPACC | NOGRPACC	The setting for group access to any group data sets protected by data set profiles defined by the user while the user is connected to the group

Data Set Profile Contents Summary

Table 55. Data Set Profile Contents: ADDSD Command

Operand of ADDSD Command:	Description
profile-name-1	The name of the data set profile (required)
/*password*	The password of a password-protected data set
GENERIC | MODEL | TAPE	A setting that indicates whether the profile is a generic profile, a model profile, or a profile that protects a tape data set
DATA('*installation-defined-data*')	Installation-defined data
OWNER(*userid* or *groupname*)	The owner of the profile
NOTIFY[(*userid*)]	The user who is to be notified whenever RACF uses this profile to deny access to a data set
UACC(*access-authority*)	The universal access authority for the data set or data sets protected by the profile
AUDIT(*access-attempts*(*audit-access-level*)...)	The type of auditing to be performed for the data set or data sets protected by the profile
FROM(*profile-name-2*)	The name of a profile that is to be used as a model
FCLASS(*profile-name-2-class*)	The class of the model profile
FGENERIC	A setting that indicates that the model profile name is to be treated as a generic name
FVOLUME(*profile-name-2-serial*)	The volume that is to be used to locate the model profile
FILESEQ(*number*)	The file sequence number of a tape data set
RETPD(*nnnnn*)	The security retention period for a tape data set
ADDCATEGORY(*category-name* ...)	The security categories to be assigned to the data set or data sets protected by the profile
SECLABEL(*security-label*)	The security label of the data set or data sets protected by the profile

Table 55. Data Set Profile Contents: ADDSD Command (continued)

Operand of ADDSD Command:	Description
SECLEVEL(*security-level*)	The security level of the data set or data sets protected by the profile
ERASE	A setting that indicates whether the data set or data sets protected by the profile are to be erased when they are scratched
LEVEL(*nn*)	An installation-defined level
NOSET | SET | SETONLY	A setting that specifies the RACF-indication status of the data set or data sets protected by the profile
UNIT(*type*)	The unit type on which the data set resides
VOLUME(*volume-serial* ...)	The volume on which the data set resides
WARNING	A setting that indicates that RACF is to issue a warning message and allow access to the data set or data sets protected by the profile even if access authority is insufficient
DFP	DFP segment information:
RESOWNER(*userid* or *groupname*)	The user ID or group name of the actual owner of the data set or data sets protected by the profile
TME	TME segment information:
ROLES(*role-access-specification* ...)	Specifies the complete list of roles and associated access levels related to this profile

Table 56. Data Set Profile Contents: Access Lists

Operand of PERMIT Command:	Description
profile-name-1	The name of the data set profile (required)
ACCESS(*access-authority*)	The access authority to be associated with the names identified on the ID operand. This operand maintains the standard access list.
ID(*name* ...|*)	The names of the users or groups to be associated with the access authority
WHEN(APPCPORT(*partner-luname* ...))	Modifies the access authority. Specifies that the identified users or groups have the specified access authority when executing commands and jobs originating from the specified partner LU.
WHEN(CONSOLE(*console-id* ...))	Modifies the access authority. Specifies that the identified users or groups have the specified access authority when executing commands and jobs originating from the specified system console.
WHEN(JESINPUT(*device-name* ...))	Modifies the access authority. Specifies that the identified users or groups have the specified access authority when entering the system through the specified JES input device.
WHEN(PROGRAM(*program-name* ...|*))	Modifies the access authority. Specifies that the identified users or groups have the specified access authority when executing the specified program.
WHEN(TERMINAL(*terminal-id* ...))	Modifies the access authority. Specifies that the identified users or groups have the specified access authority when logged on to the specified terminal.

General Resource Profile Contents Summary

Table 57. General Resource Profile Contents: RACDCERT Command

Operand of RACDCERT Command:	Description
	Base segment information (DIGTNMAP class only):

Table 57. General Resource Profile Contents: RACDCERT Command (continued)

Operand of RACDCERT Command:	Description
MAP(*data-set-name*)	Specifies a certificate name filter.
ID*(userid)*	Specifies the user ID associated with the certificate name filter.
SDNFILTER('*subject's-distinguished-name-filter*')	Specifies the significant portion of the subject's X.509 distinguished name used in a certificate name filter.
IDNFILTER('*issuer's-distinguished-name-filter*')	Specifies the significant portion of the issuer's X.509 distinguished name used in a certificate name filter.
CRITERIA(*criteria-profile-name-template*)	Indicates additional criteria used with MULTIID certificate name filters.
WITHLABEL*('label-name')*	Specifies the label of the certificate name filter.
TRUST | NOTRUST	Indicates the trust status of the certificate name filter.
	CERTDATA segment information (DIGTCERT class only):
ADD(*data-set-name*)	Assigns a digital certificate for a user ID, site or certificate authority.
NOTRUST	Indicates that the status of the certificate being added is not trusted. When a certificate is not trusted, it is not used by RACF to validate an entity.
TRUST	Indicates that the status of the certificate being added is trusted. When the certificate is trusted, it can be used by RACF to validate an entity.
HIGHTRUST	Indicates that the certificate authority is considered highly trusted. When a certificate is highly trusted, it is enabled for the same usages as a trusted certificate.
GENCERT(*request-dataset-name*)	Creates a public/private key pair and generates a digital certificate.
SUBJECTSDN	Specifies the subject's X.509 distinguished name.
SIZE(*key-size*)	Specifies the size of the private key.
NOTBEFORE(DATE(*yyyy-mm-dd*) TIME(*hh:mm:ss*))	Specifies the local date and time from which the certificate is valid.
NOTAFTER(DATE(*yyyy-mm-dd*) TIME(*hh:mm:ss*))	Specifies the local date and time after which the certificate is no longer valid.
WITHLABEL('*label-name*')	Specifies the label assigned to this certificate.
SIGNWITH(CERTAUTH LABEL('*label-name*'))	
SIGNWITH(SITE LABEL'(*label-name*'))	
SIGNWITH(LABEL'(*label-name*'))	Specifies the certificate and private key that are signing the certificate.
ICSF	Specifies that RACF should attempt to store the private key associated with this certificate in ICSF.
KEYUSAGE(HANDSHAKE)	Specifies that the key is for identification and key exchange during security handshakes, such as SSL.
KEYUSAGE(DATAENCRYPT)	Specifies that the key is for encrypting data.
KEYUSAGE(DOCSIGN)	Specifies that the key is for legally binding signatures and non-repudiation.
KEYUSAGE(CERTSIGN)	Specifies that the key is for signing other certificates and certificate revocation lists (CRLs).
ALTNAME(IP(*numeric-ip-address*))	Specifies the internet protocol address in Version 4 dotted-decimal format.
ALTNAME(DOMAIN('*internet-domain-name*'))	Specifies the internet domain name.
ALTNAME(EMAIL('*email-address*'))	Specifies the fully qualified e-mail address.
ALTNAME(URI('*universal-resource-identifier*'))	Specifies the universal resource identifier.
	CERTDATA segment information (DIGTRING class only):

Table 57. General Resource Profile Contents: RACDCERT Command (continued)

Operand of RACDCERT Command:	Description
ADDRING(*ring-name*)	Specifies the creation of a key ring for a user ID.

Table 58. General Resource Profile Contents: RDEFINE Command

Operand of RDEFINE Command:	Description
classname	The name of the class to which the resource or resources to be protected belong
profile-name-1	The name of the profile
ADDMEM(*member* ...)	The member names to be added to the profile (for resource grouping classes only)
APPLDATA('*application-data*')	Application-defined data
DATA('*installation-defined-data*')	Installation-defined data
OWNER(*userid* or *groupname*)	The owner of the profile
NOTIFY[(*userid*)]	The user who is to be notified whenever RACF uses this profile to deny access to a resource
UACC(*access-authority*)	The universal access authority for the resource or resources protected by the profile
AUDIT(*access-attempts*[(*audit-access-level*)])	The type of auditing to be performed for the resource or resources protected by the profile
FROM(*profile-name-2*)	The name of a profile that is to be used as a model
FCLASS(*profile-name-2-class*)	The class of the model profile
FGENERIC	A setting that indicates that the model profile name is to be treated as a generic name
FVOLUME(*volume-serial*)	The volume that is to be used to locate the model profile
ADDCATEGORY(*category-name* ...)	The security categories to be assigned to the resource or resources protected by the profile
SECLABEL(*seclabel-name*)	The security label of the resource or resources protected by the profile
SECLEVEL(*seclevel-name*)	The security level of the resource or resources protected by the profile
LEVEL(*nn*)	An installation-defined level
SINGLEDSN	The tape volume protected by this profile can contain only one data set (TAPEVOL class only)
TVTOC	A setting that specifies that RACF is to create a tape volume table of contents (TVTOC) when a user creates the first output data set on the tape volume (TAPEVOL class only)
TIMEZONE({E | W} *hh*[.*mm*])	The time zone in which a terminal resides (TERMINAL class only)
WHEN([DAYS(*day-info*)][TIME(*time-info*)])	The times when the terminal or terminals protected by the profile can be used to access the system (TERMINAL class only)
WARNING	A setting that indicates that RACF is to issue a warning message and allow access to the resource or resources protected by the profile even if access authority is insufficient
DLFDATA	DLFDATA segment information (DLFCLASS class only):
RETAIN(YES | NO)	A setting that indicates whether the DLF object can be retained after use
JOBNAMES(*jobname-1* ...)	The list of objects that can access the DLF objects protected by the profile
KERB	KERB segment information (REALM class only):

Table 58. General Resource Profile Contents: RDEFINE Command (continued)

Operand of RDEFINE Command:	Description
KERBNAME(*local_realm_name*)	The name of the local realm
ENCRYPT([DES | NODES] [DES3 | NODES3] [DESD | NODESD])	The key encryption options for the local realm.
MINTKTLFE(*min_ticket_life*)	The minimum ticket life for the local realm
MAXTKTLFE(*max_ticket_life*)	The maximum ticket life for the local realm
DEFTKTLFE(*def_ticket_life*)	The default ticket life for the local realm
PASSWORD(*password*)	The value of the password
SESSION	SESSION segment information (APPCLU class only):
CONVSEC	The levels of security checking performed when conversations are established with the LU protected by this profile
INTERVAL(*n*)	The maximum number of days that the session key is valid
LOCK	A setting that indicates that the profile is locked, preventing all session establishment from succeeding
SESSKEY(*session-key*)	The session key
SSIGNON	SSIGNON segment information (PTKTDATA class only):
KEYENCRYPTED(*key-value*)	Indicates that you want to encrypt the value of the secured signon application key
KEYMASKED(*key-value*)	Indicates that you want to mask the key value using the masking algorithm
STDATA	STDATA segment information (STARTED class only):
USER(*userid* | =MEMBER)	Specifies the user ID to be associated with this entry
GROUP(*groupname* | =MEMBER)	Specifies the group name to be associated with this entry
PRIVILEGED(NO | YES)	Specifies whether the started task should run with the RACF PRIVILEGED attribute
TRACE(NO | YES)	Specifies whether a message should be issued to the operator when this entry is used to assign an ID to the started task
TRUSTED(NO | YES)	Specifies whether the started task should run with the RACF TRUSTED attribute
SVFMR	SystemView segment information:
PARMNAME(*parm-name*)	Specifies the name of the parameter list associated with this application
SCRIPTNAME(*script-name*)	Specifies the name of the list of default logon scripts associated with this application
TME	TME segment information:
CHILDREN(*profile-name ...*)	Specifies the complete list of roles that inherit attributes from this role
GROUPS(*groupname ...*)	Specifies the complete list of groups that should be permitted to resources defined in this role profile
PARENT(*profile-name ...*)	Specifies the name of a role from which this role inherits attributes
RESOURCE(*resource-access-specification ...*)	Specifies the complete list of resources and associated access levels for groups defined in this role profile
ROLES(*role-access-specification ...*)	Specifies the complete list of roles and associated access levels related to this profile

Table 59. General Resource Profile Contents: Access Lists

Operand of PERMIT Command:	**Description**
profile-name-1	The name of the general resource profile (required)
CLASS(*profile-name-1-class*)	The name of the class to which the general resource profile belongs (required)
ACCESS(*access-authority*)	The access authority to be associated with the names identified on the ID operand. This operand maintains the standard access list.
ID(*name* ...|*)	The names of the users or groups to be associated with the access authority
WHEN(APPCPORT(*partner-luname* ...))	Modifies the access authority. Specifies that the identified users or groups have the specified access authority when executing commands and jobs originating from the specified partner LU.
WHEN(CONSOLE(*console-ID* ...))	Modifies the access authority. Specifies that the identified users or groups have the specified access authority when executing commands and jobs originating from the specified system console.
WHEN(JESINPUT(*device-name* ...))	Modifies the access authority. Specifies that the identified users or groups have the specified access authority when entering the system through the specified JES input device.
WHEN(SYSID(*system-ID*))	Modifies the access authority. Specifies that the identified users or groups have the specified access authority to a particular program when loading the program on the specified system.
WHEN(TERMINAL(*terminal-ID* ...))	Modifies the access authority. Specifies that the identified users or groups have the specified access authority when logged on to the specified terminal.

Appendix D. RACF/DB2 External Security Module: Authorization Checking

This appendix includes information about the RACF authorization checking through the RACF/DB2 external security module for the following DB2 objects:

B Buffer pools
C Collections
D Databases
E User-defined distinct types
F User-defined functions
J Java archives (JARs)
K Packages
M Schemas
O Stored procedures
P Application plans
R Table spaces
S Storage groups
T Tables and views
U Systems

The sections that follow outline the sequence of authorization checks that occur in the RACF/DB2 external security module to determine if the requesting user is authorized to use a particular DB2 privilege against a particular DB2 object type. If any authorization check in the sequence is successful, the privilege is granted. For examples of authorization processing in the RACF/DB2 external security module, see “Authorization Processing: Examples” on page 408.

In order to perform authorization checks, the RACF/DB2 external security module uses the values passed with the following parameters to determine the DB2 object types and privileges:

XAPLTYPE DB2 object type

XAPLPRIV DB2 privilege

Note: The sections that follow show the name of each DB2 privilege passed with the XAPLPRIV parameter. The RACF/DB2 external security module uses a numeric XAPLPRIV value. See *DB2 Administration Guide* to find the numeric value associated with each DB2 privilege name.

The profile name formats shown in this appendix are applicable if you are using multiple-subsystem scope (classification model 2). If you are using single-subsystem scope (classification model 1), the resource name does not include the DB2 subsystem name. If you are using DB2 data sharing, substitute *DB2-group-attachment-name* for *DB2-subsystem* in the profile name formats shown in this appendix.

Table Privileges

Resources: tables and views

Resource type: T

Note about SYSCTRL

The SYSCTRL administrative authority does not apply uniformly to system tables and user tables. When the authority does not apply for a user table, DB2 turns on bit 7 of the XAPLFLG1 field. If this bit is on, the RACF/DB2 external security module will bypass checking for the SYSCTRL authority. This allows RACF processing to model DB2 processing.

DB2 Privileges

ALTER

XAPLPRIV privilege name: ALTERAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.table-owner.table-name.ALTER	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

INDEX

XAPLPRIV privilege name: INDEXAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.table-owner.table-name.INDEX	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CHANGE NAME QUALIFIER

XAPLPRIV value: QUALAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

SELECT

XAPLPRIV privilege name: SELCTAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.table-owner.table-name.SELECT	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

INSERT

XAPLPRIV privilege name: INSRTAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.table-owner.table-name.INSERT	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

DELETE

XAPLPRIV privilege name: DELETAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.table-owner.table-name.DELETE	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

UPDATE

XAPLPRIV privilege name: UPDTEAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.table-owner.table-name.UPDATE	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.column.UPDATE	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

COMMENT ON, DROP

XAPLPRIV values: COMNTAUT, DROPAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

LOAD

XAPLPRIV privilege name: LOADAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.LOAD	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

REFERENCES

XAPLPRIV privilege name: REFERAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.table-owner.table-name.REFERENCES	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.ALTER	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.column.REFERENCES	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CREATE SYNONYM

XAPLPRIV privilege name: CRTSYAUT

There are no authorization checks (return code 4).

ALTER INDEX, DROP INDEX

XAPLPRIV values: ALTIXAUT, DRPIXAUT

Does the user own the index?

If so, XAPLUPRM or XAPLUCHK must match the index name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.DBADM	DSNADM

Note for DB2 Version 5 and Version 6: DB2 does not provide the database name for these requests. Therefore, the RACF/DB2 external security module cannot check the DBADM authority for the database that contains the index. As a result, permitting users to one of the following profiles covering the resource will allow users to DROP or ALTER indexes in any database. Note that DB2 Version 7 provides the database name for this request. For more information, see “DROP and ALTER INDEX Privileges” on page 400.

One of these profiles:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DROP SYNONYM

XAPLPRIV privilege name: DRPSYAUT

There are no authorization checks (return code 4).

CREATE VIEW

XAPLPRIV privilege name: CRTVUAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM
DB2-subsystem-1.DBADM *DB2-subsystem-2*.DBADM ⋮ *DB2-subsystem-n*.DBADM	DBADM DBADM ⋮ DBADM

Note: DB2 Version 5 and Version 6 do not allow DBADM authority to be used to allow a user to create views. Beginning with DB2 Version 7, DBADM authority can be used to allow a user to create views. See “CREATE VIEW Privilege” on page 401 for more information.

LOCK TABLE

XAPLPRIV privilege name: LOCKAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.table-owner.table-name.SELECT	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DROP ALIAS

XAPLPRIV privilege name: DRPALAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Any of the Table Privileges

XAPLPRIV privilege name: ANYTBAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.table-owner.table-name.REFERENCES	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.ALTER	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.INDEX	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.SELECT	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.INSERT	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.DELETE	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.UPDATE	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

CATMAINT CONVERT

XAPLPRIV privilege name: CNVRTAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

RENAME TABLE

XAPLPRIV privilege name: RNTABAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

TRIGGER
XAPLPRIV privilege name: TRIGAUT

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.table-owner.table-name.TRIGGER	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.ALTER	
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

Database Privileges

Resources: databases

Resource type: D

DB2 Administrative Authorities

DBADM
XAPLPRIV privilege name: DBAAUTH

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSADM	DSNADM

DBCTRL
XAPLPRIV privilege name: DBCTLAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DBMAINT
XAPLPRIV privilege name: DBMNTAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DB2 Privileges

CREATETAB
XAPLPRIV privilege name: CRTTBAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.CREATETAB	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CHANGE NAME QUALIFIER
XAPLPRIV privilege name: QUALAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CREATETS
XAPLPRIV privilege name: CRTTSAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.CREATETS	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DISPLAYDB

XAPLPRIV privilege name: DSPDBAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.DISPLAYDB	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.database-name.DISPLAY	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DROP

XAPLPRIV privilege name: DROPAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.DROP	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

IMAGCOPY, MERGECOPY, MODIFY RECOVERY, QUIESCE

XAPLPRIV values: IMCOPAUT, MERGEAUT, MODAUT, QUIESAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.IMAGCOPY	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM

One of these profiles:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

LOAD

XAPLPRIV privilege name: LOADAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.LOAD	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

REORG

XAPLPRIV privilege name: REORGAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.REORG	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

RECOVERDB, REPORT

XAPLPRIV values: RECDBAUT, REPRTAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.RECOVERDB	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

REPAIR, RUN REPAIR UTILITY

XAPLPRIV values: REPARAUT, DIAGAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.REPAIR	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBCTRL	DSNADM

One of these profiles:	In class:
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

STARTDB

XAPLPRIV privilege name: STARTAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.STARTDB	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

RUN CHECK UTILITY, STATS

XAPLPRIV values: CHECKAUT, STATSAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.STATS	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

STOPDB

XAPLPRIV privilege name: STOPAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.STOPDB	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

TERM UTILITY

XAPLPRIV privilege name: TERMAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

REPAIR DBD

XAPLPRIV privilege name: RDBDAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

TERM UTILITY ON DATABASE

XAPLPRIV privilege name: TERMDAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM

Plan Privileges

Resources: application plans

Resource type: P

BIND

XAPLPRIV privilege name: BINDAUT

Does the user own the plan?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLOWNQ parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.plan-name.BIND	MDSNPN or GDSNPN
DB2-subsystem.owner.BINDAGENT	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

EXECUTE

XAPLPRIV privilege name: CHKEXEC

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.plan-name.EXECUTE	MDSNPN or GDSNPN
DB2-subsystem.SYSADM	DSNADM

Package Privileges

Resources: packages

Resource type: K

EXECUTE

XAPLPRIV privilege name: CHKEXEC

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.collection-ID.package-ID.EXECUTE	MDSNPK or GDSNPK
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSADM	DSNADM

BIND

XAPLPRIV privilege name: BINDAUT

Does the user own the package?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.collection-ID.package-ID.BIND	MDSNPK or GDSNPK
DB2-subsystem.owner.BINDAGENT	MDSNSM or GDSNSM
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

COPY

XAPLPRIV privilege name: COPYAUT

Does the user own the package?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.collection-ID.package-ID.COPY	MDSNPK or GDSNPK
DB2-subsystem.owner.BINDAGENT	MDSNSM or GDSNSM
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

All Package Privileges (PACKADM or SYSADM)

XAPLPRIV privilege name: ALLPKAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSADM	DSNADM

All Package Privileges (PACKADM, SYSADM, or SYSCTRL)

XAPLPRIV privilege name: SUBPKAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.PACKADM	DSNADM
The user has authority to all collections.	
DB2-subsystem.collection-ID.PACKADM	DSNADM
The user has authority to *collection-ID*.	
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DROP

XAPLPRIV privilege name: DROPAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Buffer Pool Privileges

Resources: buffer pools

Resource type: B

USE

XAPLPRIV privilege name: USEAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.buffer-pool-name.USE	MDSNBP or GDSNBP
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Collection Privileges

Resources: collections

Resource type: C

DB2 Privileges

CREATE IN

XAPLPRIV privilege name: CRTINAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.collection-ID.CREATEIN	MDSNCL or GDSNCL
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DB2 Administrative Authorities

PACKADM

XAPLPRIV privilege name: PKADMAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSADM	DSNADM

Table Space Privileges

Resources: table spaces

Resource type: R

DB2 Privileges

USE

XAPLPRIV privilege name: USEAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.tablespace-name.USE	MDSNTS or GDSNTS
DB2-subsystem.database-name.DBADM The user has DATABASE ADMINISTRATOR authority for the database named in CHKCRATR.	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DROP, ALTER

XAPLPRIV privilege name: DROPAUT, ALTERAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Storage Group Privileges

Resources: storage groups

Resource type: S

USE

XAPLPRIV privilege name: USEAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.storage-groupname.USE	MDSNSG or GDSNSG
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DROP, ALTER

XAPLPRIV privilege name: DROPAUT ALTERAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

System Privileges

Resources: systems

Resource type: U

DB2 Administrative Authorities

SYSADM
XAPLPRIV privilege name: SYSAAUTH

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.SYSADM	DSNADM

SYSCTRL
XAPLPRIV privilege name: SYSCAUTH

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

SYSOPR
XAPLPRIV privilege name: SYSOAUTH

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DB2 Privileges

BINDADD
XAPLPRIV privilege name: BINDAAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.BINDADD	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

RECOVER BSDS
XAPLPRIV privilege name: CHKBSDS

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.BSDS	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CREATEDBA

XAPLPRIV privilege name: CRTDBAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.CREATEDBA	MDSNSM or GDSNSM
DB2-subsystem.CREATEDBC	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CREATEDBC

XAPLPRIV privilege name: CRTDCAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.CREATEDBC	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CREATESG

XAPLPRIV privilege name: CRTSGAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.CREATESG	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DISPLAY, DISPLAY BUFFERPOOL, DISPLAY PROCEDURE

XAPLPRIV values: CHKDISPL, CHKDSPBP, DSPPRAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.DISPLAY	MDSNSM or GDSNSM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

RECOVER INDOUBT

XAPLPRIV privilege name: CHKRECOV

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.RECOVER	MDSNSM or GDSNSM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

STOPALL

XAPLPRIV privilege name: CHKSUBSY

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.STOPALL	MDSNSM or GDSNSM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

TRACE

XAPLPRIV privilege name: CHKTRACE

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.TRACE	MDSNSM or GDSNSM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

STOSPACE UTILITY

XAPLPRIV privilege name: STOAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.STOSPACE	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CANCEL|START|STOP DDF, DISPLAY|START|STOP RLIMIT, START|STOP PROCEDURE

XAPLPRIV values: CHKSTART, CHKSTOP, CHKDSPL, CHKDDF, STRPRAUT, STPPRAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

MONITOR1

XAPLPRIV privilege name: MON1AUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.MONITOR1	MDSNSM or GDSNSM
DB2-subsystem.MONITOR2	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

MONITOR2

XAPLPRIV privilege name: MON2AUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.MONITOR2	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CREATEALIAS

XAPLPRIV privilege name: CRTALAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.CREATEALIAS	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

USE ARCHIVE LOG

XAPLPRIV privilege name: ARCHAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.ARCHIVE	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

BINDAGENT
XAPLPRIV privilege name: BNDAGAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.owner.BINDAGENT	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

SET ARCHIVE
XAPLPRIV privilege name: SARCHAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.ARCHIVE	MDSNSM or GDSNSM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DISPLAY ARCHIVE
XAPLPRIV privilege name: DARCHAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.DISPLAY	MDSNSM or GDSNSM
DB2-subsystem.ARCHIVE	MDSNSM or GDSNSM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CREATETMTAB
XAPLPRIV privilege name: CRTTMAUT

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.CREATETMTAB	MDSNSM or GDSNSM
DB2-subsystem.CREATETAB	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

User-Defined Distinct Type Privileges

Resources: user-defined distinct types

Resource type: E

USAGE

XAPLPRIV privilege name: USAGEAUT

Does the user own the user-defined distinct type?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.schema-name.type-name.USAGE	MDSNUT or GDSNUT
DB2-subsystem.SYSADM	DSNADM

User-Defined Function Privileges

Resources: user-defined functions

Resource type: F

DISPLAY

XAPLPRIV privilege name: DISPAUT

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the user-defined function?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.schema-name.function-name.DISPLAY	MDSNUF or GDSNUF
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

EXECUTE

XAPLPRIV privilege name: CHKEXEC

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.schema-name.function-name.EXECUTE	MDSNUF or GDSNUF
DB2-subsystem.SYSADM	DSNADM

START

XAPLPRIV privilege name: STRTAUT

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the user-defined function?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

STOP

XAPLPRIV privilege name: STPAUT

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the user-defined function?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Stored Procedure Privileges

Resources: stored procedures

Resource type: O

DISPLAY

XAPLPRIV privilege name: DISPAUT

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the stored procedure?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
*DB2-subsystem.schema-name.procedure-name.*DISPLAY	MDSNSP or GDSNSP
*DB2-subsystem.*SYSOPR	DSNADM
*DB2-subsystem.*SYSCTRL	DSNADM
*DB2-subsystem.*SYSADM	DSNADM

EXECUTE

XAPLPRIV privilege name: CHKEXEC

The user must have sufficient authority to:

One of these profiles:	In class:
*DB2-subsystem.schema-name.procedure-name.*EXECUTE	MDSNSP or GDSNSP
*DB2-subsystem.*SYSADM	DSNADM

START

XAPLPRIV privilege name: STRTAUT

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the stored procedure?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
*DB2-subsystem.*SYSOPR	DSNADM
*DB2-subsystem.*SYSCTRL	DSNADM
*DB2-subsystem.*SYSADM	DSNADM

STOP

XAPLPRIV privilege name: STPAUT

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the stored procedure?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Schema Privileges

Resources: schemas

Resource type: M

ALTERIN

XAPLPRIV privilege name: ALTINAUT

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the object?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.schema-name.object-name.ALTERIN	MDSNSC or GDSNSC
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

COMMENT ON

XAPLPRIV privilege name: COMNTAUT

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the object?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
*DB2-subsystem.schema-name.object-name.*ALTERIN	MDSNSC or GDSNSC
*DB2-subsystem.*SYSCTRL	DSNADM
*DB2-subsystem.*SYSADM	DSNADM

CREATEIN

XAPLPRIV privilege name: CREINAUT

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOBJN parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
*DB2-subsystem.schema-name.*CREATEIN	MDSNSC or GDSNSC
*DB2-subsystem.*SYSCTRL	DSNADM
*DB2-subsystem.*SYSADM	DSNADM

DROPIN

XAPLPRIV privilege name: DRPINAUT

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the object?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
*DB2-subsystem.schema-name.object-name.*DROPIN	MDSNSC or GDSNSC
*DB2-subsystem.*SYSCTRL	DSNADM
*DB2-subsystem.*SYSADM	DSNADM

CHANGE NAME QUALIFIER

XAPLPRIV privilege name: QUALAUT

DB2 Privileges

The user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Java Archive (JAR) Privileges

Resources: Java archives (JARs)

Resource type: J

USAGE

XAPLPRIV privilege name: USAGEAUTJ

Does the user own the Java archive (JAR)?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these profiles:	In class:
DB2-subsystem.schema-name.JAR-name.USAGE	MDSNJR or GDSNJR
DB2-subsystem.SYSADM	DSNADM

Appendix E. Security for System Data Sets

This appendix contains some guidelines for defining UACC values and security labels for system data sets.

Table 60 lists the UACC values that should be assigned to many of the system data sets. Note that your security policy may require a UACC of NONE for some data sets. For example, you can specify UACC(NONE) for macro libraries if you give READ access to programmers who need to assemble or compile programs that use those libraries. For a discussion of system data sets, see “Protecting DASD System Data Sets” on page 172.

Table 60 also lists the security labels that the system data sets must have on an evaluated B1 system. In general, in a B1 environment, you should assign a SECLABEL of SYSLOW to system data sets. If a data set is not listed in the table, it should be assigned a SECLABEL of SYSLOW.

You should consider creating a generic profile to protect system data sets, as follows:

```
ADDSD 'SYS1.*' UACC(NONE) SECLABEL(SYSHIGH)
```

Specifying a UACC of NONE for the SYS1.* profile protects any system data sets that are added to the system by new products. If new system data sets need a UACC higher than NONE or a SECLABEL of SYSLOW, you can create more specific profiles for them.

You should also create specific profiles for particular data sets (or groups of data sets, such as SYS1.DUMP*xx* data sets), using the information in Table 60.

For any data set that is listed with a UACC of READ or higher, you should consider creating an entry in the global access checking table. For more information, see “Setting Up the Global Access Checking Table” on page 206.

For system data sets that are listed in the table with a UACC higher than NONE, you might prefer to specify UACC(NONE) and create an access list entry of ID(*) ACCESS(*access-authority*). This entry prevents restricted users and users who are not defined to RACF from accessing the data sets.

Restricted users enter the system with a user ID that is defined with the RESTRICTED attribute, and may be shared by many users. Restricted users are prevented from gaining access to protected resources through the global access checking table, UACC, or the ID(*****) entry on the access list. User IDs defined with the RESTRICTED attribute must be specifically authorized with sufficient authority on the access list of any protected resource they must access. Therefore, to allow restricted users to access any data set listed with UACC of READ or higher in Table 60, each user ID with the RESTRICTED attribute must be specifically authorized at the level of access indicated by the UACC value.

Table 60. UACC Values and B1 Security Labels for System Data Sets

Data Set	UACC	B1 Security Label	Comments
Master catalog	READ	SYSNONE	
User catalogs	UPDATE	SYSNONE	

Table 60. UACC Values and B1 Security Labels for System Data Sets (continued)

Data Set	UACC	B1 Security Label	Comments
Checkpoint data sets	NONE	SYSHIGH	
Distribution library data sets	NONE	SYSLOW	
ISPF panel libraries	READ	SYSLOW	Panel definitions, skeletons, CLISTs, and so forth. Specify UACC(NONE) if access must be restricted.
JES2 offload data sets	NONE	SYSHIGH	
Load libraries	READ	SYSLOW	
Page data sets	NONE	SYSHIGH	Include PLPA, common, and local page data sets. See *z/OS MVS Initialization and Tuning Guide*.
PSF secure font data sets	NONE	SYSLOW	
PSF secure overlay data sets	NONE	SYSLOW	
PSF secure page segment data sets	NONE	SYSLOW	
RMF data sets	NONE	SYSHIGH	VSAM data sets.
Security definitions data sets	NONE	SYSLOW	
SMP data sets	NONE	SYSLOW	
Swap data sets	NONE	SYSHIGH	
SYS1.AMACLIB	READ	SYSLOW	
SYS1.AMODGEN	READ	SYSLOW	
SYS1.ASAMPLIB	READ	SYSLOW	
SYS1.BRODCAST (in a non-B1 environment)	UPDATE	SYSLOW	
SYS1.BRODCAST (in a B1 environment)	READ	SYSLOW	
SYS1.CMDLIB	READ	SYSLOW	
SYS1.DAE	NONE	SYSHIGH	
SYS1.DUMP*xx*	NONE	SYSHIGH	See *z/OS MVS Initialization and Tuning Reference*.
SYS1.HELP	READ	SYSLOW	TSO online help.
SYS1.IMAGELIB	NONE	SYSLOW	
SYS1.JESPARM	NONE	SYSLOW	
SYS1.JES3LIB	READ	SYSLOW	
SYS1.LINKLIB	READ	SYSLOW	
SYS1.LOGREC	NONE	SYSLOW	
SYS1.LPALIB	READ	SYSLOW	UACC can be NONE or READ depending on installation security policy.
SYS1.MACLIB	READ	SYSLOW	

Table 60. UACC Values and B1 Security Labels for System Data Sets (continued)

Data Set	UACC	B1 Security Label	Comments
SYS1.MANx	NONE	SYSHIGH	SMF data sets. See *z/OS MVS Initialization and Tuning Reference*.
SYS1.MIGLIB	READ	SYSLOW	
SYS1.MODGEN	READ	SYSLOW	
SYS1.NUCLEUS	READ	SYSLOW	
SYS1.OVERLIB	READ	SYSLOW	
SYS1.PARMLIB	READ	SYSLOW	UACC should be NONE if any members contain passwords, or other sensitive information, such as the ACBPW password in the TSOKEYxx member.
SYS1.PROCLIB	READ	SYSLOW	
SYS1.RACF	NONE	SYSLOW	Can have other names in your installation.
SYS1.SAMPLIB	READ	SYSLOW	
SYS1.STGINDEX	NONE	SYSHIGH	
SYS1.SVCLIB	NONE	SYSLOW	
SYS1.TELCMLIB	READ	SYSLOW	
SYS1.UADS	NONE	SYSLOW	
SYS1.VTOCIX.___	NONE	SYSLOW	
SYS1.VVDS.___	NONE	SYSLOW	
SYS1.VTAMLIB	READ	SYSLOW	
SYS1.VTAMLST	NONE	SYSLOW	
Trace data sets	NONE	SYSHIGH	
User dump data sets	NONE	User's SECLABEL	

Appendix F. Debugging Problems in the RACF Database

This appendix explains how to debug the profile definitions in the RACF database and the current RACF options. It also discusses RACF authorization checking and refresh timing.

This appendix describes how to debug problems with your profile definitions. It is designed to help you determine how your current RACF options and profiles work for you. For example, if users have access to resources that they should not have, or if users do not have access to resources that they should have, this section might be helpful in determining how to correct the problem.

Note: If you have a problem that you suspect is caused by RACF, and not by your use of RACF, see for information on correcting the problem.

Checklist: Resolving Problems When Access Is Denied Unexpectedly

When a user or job requires access to a protected resource, and RACF denies the requested access, you must often analyze the problem before deciding what action to take. Here are some basic steps to take when analyzing access problems:

- First, get the complete text of the ICH408I message that RACF issued when denying the access. Also, take note of any other messages that were issued. The ICH408I message often indicates what profile RACF checked, and what class the profile was in, when denying access. For a description of the ICH408I message, see *z/OS SecureWay Security Server RACF Messages and Codes*.

  Note: List the profile you believe should have given access. If it contains an *, %, or &, make sure it is also followed by a **G**. If you do not have the **G**, it is not generic and would not be granting any access.
- If the ICH408I message indicates that access was denied because of a revoked user ID, you may want to resume that user ID. Check if the user ID is associated

with the started procedure. If there was a user ID associated with the started procedure, this started procedure could not have begun successfully. After you resume the user ID, you must restart the started procedure or re-IPL.

- If the ICH408I message indicates that access was denied because of a profile, check the profile listing to make sure the user or job should have access. You should check not only the UACC and access lists, but the security classification of the resource profile and the user. Also, please note that installation exits (both RACF exits and certain exits in products that call RACF, such as JES) can affect a user's access to resources. To check the user's access to the resource, ask the user who had the problem to list the profile protecting the resource.
  - For data sets, the user can issue the LISTDSD command.
  - For general resources, the user can issue the RLIST command.

  In the listing, have the user check the YOUR ACCESS field.
- If the user cannot issue the LIST command, do it yourself. In the listing supplied by RACF, the following fields can, by themselves, deny access to a user:
  - The security level or security category, or both
  - The security label
  - The standard access list (ID and ACCESS)
  - The universal access authority (UACC)
- If the profile listing indicates that the user or job should have access, and the profile is in a class for which SETROPTS RACLIST processing or SETROPTS GENLIST processing is in effect, make sure that any in-storage profiles are refreshed by doing the following:
  - If the resource is protected by a generic profile in a class that is not RACLISTed or GENLISTed, ask the user to log off and log on again, or resubmit the job. This refreshes the user's copy of the profile.
  - Issue the SETROPTS GENERIC(*classname*) REFRESH or SETROPTS RACLIST(*classname*) REFRESH command again.

    For information about SETROPTS REFRESH processing on shared systems, see "Refreshing Shared Systems (REFRESH Option)" on page 135.
- You can use audit records to gather information that you would not otherwise have. In particular, you can request that audit records be generated for all accesses to protected resources, or for only failed accesses. You can also request that audit records be kept for a particular user. With the auditing in effect, you can use the RACF report writer to view the access requests associated with the access requests.

  Note: In some cases (such as some resources in the OPERCMDS class), a RACROUTE request from a resource manager can include a "log string", which is a string of characters to be placed in the SMF record if the access is audited. The log string can be useful in determining what kind of action the user was taking. For example, the log string might include the exact operator command, as the operator issued it.

Checklist: Resolving Problems When Access Is Allowed Incorrectly

You may see occurrences when a user or job obtains access to a protected resource and you believe that the user should not have that access. There are many reasons why this could happen. The following checklist can eliminate some of them:

- Make sure that RACF is active, and that message ICH520I has been issued for this IPL. (To see if RACF is active, issue a RACF command, such as the LISTUSER command. If RACF is not active, the command will fail.)

Note: Even though RACF is active, resource managers for which external security is optional (such as CICS) might not be using RACF. You must ensure that such resource managers are calling RACF. Also, there may be other options that control which general resource classes are protected by RACF. Therefore, you must make sure that the resource manager is calling RACF for the particular resource class. For CICS, you must ensure that the particular CICS region is using external security. For information specifically related to CICS, see *CICS RACF Security Guide*.

- If the resource is a general resource (in other words, not protected by a profile in the DATASET class), make sure that the general resource class is active. For example, for tape volumes, make sure that the TAPEVOL class is active. To do this, issue the SETROPTS LIST command.
- Check for a global access checking table entry for the resource. An entry in the global access checking table can allow access for all users, except those with the RESTRICTED attribute, when a profile protecting the resource would deny access. For example, for data sets, enter:

  ```
  RLIST GLOBAL DATASET
  ```

  Note: Do not ignore the presence of entries containing &RACGPID or &RACUID.
- If the profile is a generic profile, use the SETROPTS LIST command to ensure that generic profile checking is in effect for the class.
- Make sure you know which profile is actually protecting the resource. For example, a more specific profile might actually be used instead of the generic profile you believe protects the resource. The more specific profile might grant the user the access. To do this, issue the LISTDSD and RLIST command, specifying the resource name. For the LISTDSD command, if you receive a message indicating that no profile is found, issue the command again with the GENERIC operand to check for any generic profiles that might protect the resource.
- Check the user's access to the resource. You can do this in two ways:
  - Ask the user to list the profile protecting the resource. For example, the user can issue the LISTDSD and RLIST command, specifying the resource name. For the LISTDSD command, if the user receives a message indicating that no profile is found, have the user issue the command again with the GENERIC operand to check for any generic profiles that might protect the resource.

    Have the user check the YOUR ACCESS field in the profile listing. If this field indicates that the user or job should have access, use the steps described in "Authorizing Access to RACF-Protected Resources" on page 634 to analyze the profile for reasons why the user or job has that access.
  - If the user cannot issue the LIST command, do it yourself. In the listing supplied by RACF, the following fields can, by themselves, allow access to a user:
    - For data sets and minidisks, the high-level qualifier
    - The standard access list
    - The conditional access list
    - UACC
    - WARNING

    If list-of-groups processing is in effect on your system, check to see if a group of which the user is a member is in the access list. Check both the standard access list and the conditional access list. Also, note that installation exits (both RACF exits and certain exits in products that call RACF, such as JES) can affect a user's access to resources.

- If your analysis of the protecting profile shows that the user should not have access, continue with the following checks.
- If the SETROPTS RACLIST processing or SETROPTS GENLIST processing is in effect, make sure that any in-storage profiles are refreshed.
  - If the resource is protected by a generic profile in a class that is not RACLISTed or GENLISTed, ask the user to log off and log on again, or resubmit the job. This refreshes the user's copy of the profile.
  - Issue the SETROPTS GENERIC(*classname*) REFRESH or SETROPTS RACLIST(*classname*) REFRESH command again.

    For information about SETROPTS REFRESH processing on shared systems, see "Refreshing Shared Systems (REFRESH Option)" on page 135.
- You can use audit records to gather information that you would not otherwise have. In particular, you can request that audit records be generated for all accesses to protected resources, or for only successful accesses. You can also request that audit records be kept for a particular user. With the auditing in effect, you can use the RACF report writer to view the access requests associated with the access requests.

  Note: In some cases (such as some resources in the OPERCMDS class),a RACROUTE request from a resource manager can include a "log string", which is a string of characters to be placed in the SMF record if the access is audited. The log string can be useful in determining what kind of action the user was taking. For example, the log string might include the exact operator command, as the operator issued it.

When Changes to Data Set Profiles Take Effect

If a user is currently using a data set, changing the data set profile protecting the data set may not affect the user's current access until that user logs on again.

The change affects the user's access immediately in the following cases:

- If the user is not logged on. You can check to see if a user is logged on with the TSO STATUS command:

  ```
  STATUS userid
  ```

  If the user is logged on, the system displays a message indicating that a job with the letters TSU in it is executing.
- If the user is logged on and has not yet opened the data set or a data set protected by the same generic profile (for example, by browsing or editing).

If the user is logged on and has opened the data set, and you change his access, two situations could occur:

- If the profile is a discrete profile, the user's access changes after closing the data set.
- If the profile is a generic profile, the user's access changes after *one* of the following occurs:
  - The user issues the LISTDSD command as follows:

    ```
    LISTDSD DATASET(dataset-protected-by-the-profile) GENERIC
    ```

    This places a fresh copy of the profile in the user's address space.
  - A SETROPTS GENERIC(DATASET) REFRESH is issued on the system the user is logged on to or from another system in the RACF sysplex data sharing group, if RACF is enabled for sysplex communication.

- The user references more than four data sets with different high-level qualifiers, and the data sets are protected by generic profiles.
- The user logs off and then logs back on.

Authorization Checking for RACF-Protected Resources

This section includes the following information:

When Authorization Checking Takes Place and Why

When a user requests access to a RACF-protected resource (such as a data set), the resource manager issues the RACROUTE macro with REQUEST=AUTH specified (or the RACHECK macro[9]). For ease of reference, this section calls such a request a RACF authorization request.

Based on the specifications on the RACF authorization request, RACF determines whether the requesting user is authorized to access the resource.

- If the user is authorized to the resource, RACF returns a "successful" return code to the resource manager. The resource manager then allows the request to complete.
- If the user is not authorized to the resource, RACF returns an "unauthorized" return code to the resource manager. The resource manager then fails the request.

  RACF issues a message indicating that the resource is not protected.
- If the resource is not protected (for example, if no profile exists for it), RACF returns the default return code for the class.

  For general resource classes, the default return code is the "not protected" return code, unless otherwise specified in the class descriptor table (CDT) entry for the class.

  For the DATASET class, the default return code is the "not protected" return code, unless the SETROPTS PROTECTALL(FAILURES) option is in effect, in which case the default return code is the "not authorized" return code.

  If the "not protected" return code is issued, the resource manager then either fails or allows the request. Most resource managers allow the request.

  RACF issues a message indicating that the resource is not protected.

Notes:

1. SMF log records or messages may be generated, depending on the options in effect and whether RACF granted or denied access to the resource.
2. When checking authorization for a directed command, RACF uses the authorization of the target user ID, not the issuing user ID.

9. If the RACHECK macro is issued instead of RACROUTE, authorization processing begins at step 6 on page 634.

Authorizing Access to RACF-Protected Resources

To perform authorization checking for RACF-protected resources, RACF makes the following checks. RACF stops processing when the request is granted or denied.

For a pictorial view of the following steps, see Figure 70 on page 640 through Figure 72 on page 642.

Note: The following steps do not apply to accessing z/OS UNIX resources. For more information, see *z/OS UNIX System Services Planning*.

1. The SAF router exits can grant or deny access before RACF authorization processing occurs.

   Note: Before master scheduler initialization (MSI),this is the ICHRTX01 exit routine. After master scheduler initialization (MSI), this is the ICHRTX00 exit routine.
2. If the entry for the class in the RACF router table is missing or has NONE specified, RACF returns the “not protected” return code. This also occurs if the caller specified the REQSTOR and SUBSYS operands on the RACROUTE macro, did not also specify DECOUPL=YES or give the REQSTOR and SUBSYS information in the RACF router table entry.
3. If RACF is not active, RACF returns the “not protected” return code.
4. For general resource classes, if the class of the resource is not active, RACF returns the “not protected” return code.
5. If the RACF class must be RACLISTed, as specified in the class descriptor table (CDT), but is not currently RACLISTed, RACF returns the “not protected” return code.
6. If the user requesting access is “trusted” or “privileged”, RACF grants the request. See the following:
   - If the user has the trusted attribute, RACF grants the request (unless the CSA or PRIVATE operand was specified on the authorization request). Such requests can be audited only by using the LOGOPTIONS operand on the SETROPTS command (which audits access requests issued by all users).
   - If the user has the privileged attribute, RACF grants the request (unless the CSA or PRIVATE operand was specified on the authorization request). Such requests cannot be audited.
7. RACF invokes the naming convention table if:
   - The naming convention routine exists
   - The resource being checked is a CLASS data set

   The naming convention table can continue REQUEST=AUTH processing or deny the request.
8. The REQUEST=AUTH preprocessing exit (ICHRCX01) can grant or deny the request.
9. If the requesting user has a default user token (created by SAF), RACF denies the request.
10. If SETROPTS MLQUIET is in effect (see “Quiescing RACF Activity (MLQUIET Option)” on page 137), RACF denies the request unless the user has the SPECIAL attribute, has the privileged or trusted attribute, or is logged on at a system console.
11. If the user ID in the RTOKEN for the resource matches the user ID in the UTOKEN for the user making a request, RACF grants the request.

This allows a user to cancel one of the user's own jobs using the TSO CANCEL command without being affected by the user's current security label.

12. If global access checking is active for the class, RACF searches the global access table (unless the CSA or PRIVATE operand was specified on the authorization request). If RACF finds a matching entry that allows access to the resource, RACF grants the request for all users, except those with the RESTRICTED attribute.
13. RACF looks for a profile in storage or in the RACF database. If no profile is found that protects the resource, RACF returns the default return code of the class. See the entry for the class in the class descriptor table (CDT) described in *z/OS SecureWay Security Server RACF Macros and Interfaces*.) Specifically, no profile is found in the following cases:
    - Profiles for the class exist in the user's storage or in a data space, but no profile matches the resource name.
    - Profiles for the class do not exist in the user's storage, in a data space, or in the RACF database.

    Note: If you expect generic profiles to be used by RACF authorization checking, list their profile names using the SEARCH command. If the profile names listed by the SEARCH command are *not* followed by (G), RACF does not treat them as generic profiles. To recover, take the following steps:

    a. Issue SETROPTS NOGENERIC(*classname*).
    b. Issue SETROPTS NOGENCMD(*classname*).
    c. Delete the profiles. (If the profiles have complicated specifications, such as long access lists, you might wish to define "dummy" profiles modeled on them before deleting them. Specify the FROM operand on the RDEFINE command.
    d. Issue SETROPTS GENERIC(*classname*).
    e. Define the profiles again.
14. If your installation has activated the SECLABEL class, RACF performs security label authorization checking. For a complete description, see "Security Label Authorization Checking" on page 647. If security label authorization checking succeeds, RACF authorization checking continues with step 16 on page 636. To check what your system configuration should be, see "B1 Configuration Requirements" on page 12.
15. If the SECLABEL class is *not* active, the SECDATA class is active, and the requested resource has a security level or security category specified, RACF makes two checks in the sequence described below.

    a. RACF compares the security level (SECLEVEL) in the user profile with the security level in the resource profile. If the resource has a higher security level than the user, or if the user has no security level, RACF denies the request.

       For a terminal session, RACF uses the lower of the user's SECLEVEL and the terminal's SECLEVEL when authorizing access to a resource. For example, if the user has a SECLEVEL of 100 and the terminal has a SECLEVEL of 50, RACF uses the terminal's SECLEVEL during authorization checking. Thus, in this case, the user cannot access, through the terminal, any resource with a SECLEVEL greater than 50. (If the terminal is not defined to RACF or is defined without a SECLEVEL, RACF uses the user's SECLEVEL to determine the resources that can be accessed.)

If the security level check passes, authorization checking continues with the following check.

b. RACF compares the list of security categories in the user profile with the list of security categories in the resource profile. If the resource profile contains a security category that is not in the user's profile, RACF denies the request.

   Unlike the security level check, RACF uses *only* the user's security category list for a terminal session.

If both checks succeed, RACF continues authorization checking with step 16.

16. If users attempt to access their own resources, RACF grants the request. For example:
    - For tape and DASD data sets, if the user ID of the requesting user is the high-level qualifier of the data set name, RACF grants the request.
    - For spool data sets, if the JESSPOOL class is active, RACF compares the user ID and node of the requester with the user ID and node of the creator of the spool data set (using the security token). If the user IDs match, RACF grants the request.
17. RACF checks the user's access authority in the standard access list. If the user is in the list and if the specified access authority is sufficient to allow access, RACF grants the request. If the user is in the list and if the specified access authority is *less than* the requested access, RACF continues processing at step 22 on page 637 (conditional access list checking). This prevents access based on ID(*****), UACC, or the OPERATIONS attribute.

    This could happen if, for example, user JOE requests UPDATE access, and the standard access list includes ID(JOE) ACCESS(READ).
18. RACF determines whether the user has access to the resource because the user is a member of a group and the group is on the standard access list. Which group is used depends on whether list-of-groups processing is in effect. (List-of-groups processing is in effect if the SETROPTS command has been issued with the GRPLIST operand.) RACF determines which group to use according to the following rules:
    - If list-of-groups processing is not in effect, RACF uses only the user's current connect group.
    - If list-of-groups processing is in effect, RACF finds all of the groups to which the user is connected that are also in the access list. Of these groups, RACF uses the group that has the highest access authority to the resource. (For example, assume that a user is a member of groups A, B, and C. If group A has NONE access authority, group B has READ access authority, and group C has UPDATE access authority, RACF uses group C to determine the user's access.)

      If the highest access authority is sufficient to allow the requested access, RACF grants the request. If the highest group that was found in the list does not have the requested authority, RACF continues processing at step 22 on page 637 (conditional access list checking). This prevents access based on ID(*****), UACC, or the OPERATIONS attribute.
19. If a user ID of ***** is found on the standard access list, the current user is defined to RACF without the RESTRICTED attribute, and the access authority granted to ***** is:
    - Sufficient to allow the requested access, RACF grants the request.

- *Not* sufficient to allow the requested access, RACF continues processing at step 21 (OPERATIONS attribute checking).

20. If the universal access authority (UACC) for the resource provides sufficient access authority and the requesting user is not defined with the RESTRICTED attribute, RACF grants the request.
21. If the requesting user has the OPERATIONS attribute (or group-OPERATIONS if the resource is within the scope of that group) and OPERATIONS access is allowed for the class, RACF grants the request.
22. RACF checks the user's access authority in the conditional access list specified with WHEN(TERMINAL), WHEN(CONSOLE), WHEN(APPCPORT), or WHEN(JESINPUT). If the user is in the list, if the user meets the specified condition (such as logged on at the specified terminal), and if the specified access authority is sufficient to allow access, RACF grants the request.
23. RACF determines whether the user has access to the resource because the user is a member of a group that meets a condition specified on the conditional access list specified with WHEN(TERMINAL), WHEN(CONSOLE), WHEN(APPCPORT), or WHEN(JESINPUT). Which group is used depends on whether list-of-groups processing is in effect. (List-of-groups processing is in effect if the SETROPTS command has been issued with the GRPLIST operand). RACF determines which group to use according to the following rules:
    - If list-of-groups processing is not in effect, RACF uses only the user's current connect group.
    - If list-of-groups processing is in effect, RACF finds all of the groups to which the user is connected that are also in the access list. Of these groups, RACF uses the group that has the highest access authority to the resource. (For example, assume that a user is a member of groups A and B. If A has READ access authority and B has UPDATE access authority, RACF uses group B to determine the user's access.)

    If the group to be used according to the preceding rules has sufficient access authority to allow the requested access, RACF grants the request. If none of the user's groups has sufficient authority, RACF does not grant the request, and continues with the next step.
24. If a user ID of ***** is found on the conditional access list specified with WHEN(TERMINAL), WHEN(CONSOLE), WHEN(APPCPORT), or WHEN(JESINPUT), and if the current user is defined to RACF without the RESTRICTED attribute, and if the current user meets the specified condition (such as logged on at the specified terminal), and the access authority granted to ***** is sufficient to allow the requested access, RACF grants the request.
25. RACF checks the user's access authority in the conditional access list specified with WHEN(PROGRAM). If the user is in the list, if the user meets the specified condition (such as running the specified program), and if the specified access authority is sufficient to allow access, RACF grants the request.

    Note: For DASD data sets, if program control is active anda controlled program is executing, RACF performs authorization checking for program access to data sets. If the user/program combination is in the conditional access list with sufficient authority to allow access to the data sets, RACF grants the request. For more information, see "Authorization Checking for Program Access to Data Sets" on page 258.
26. RACF determines whether the user has access to the resource because the user is a member of a group that meets a condition specified on the

conditional access list (such as running a specified program). Which group is used depends on whether list-of-groups processing is in effect. (List-of-groups processing is in effect if the SETROPTS command has been issued with the GRPLIST operand.) RACF determines which group to use according to the following rules:

- If list-of-groups processing is not in effect, RACF uses only the user's current connect group.
- If list-of-groups processing is in effect, RACF finds all of the groups to which the user is connected that are also in the access list. Of these groups, RACF uses the group that has the highest access authority to the resource. (For example, assume that a user is a member of groups A and B. If A has READ access authority and B has UPDATE access authority, RACF uses group B to determine the user's access.)

If the group to be used according to the preceding rules has sufficient access authority to allow the requested access, RACF grants the request. If the group is in the list and if the specified access authority is NONE, RACF denies the request.

27. If a user ID of ***** is found on the conditional access list specified with WHEN(PROGRAM), and if the current user is defined to RACF without the RESTRICTED attribute, and if the current user meets the specified condition (such as logged on at the specified terminal or running the specified program), and the access authority granted to ***** is sufficient to allow the requested access, RACF grants the request.
28. If the WARNING flag is ON in the profile (set using the WARNING operand on the ADDSD, ALTDSD, RDEFINE, or RALTER command), RACF grants the request.
29. For access to uncataloged data sets, if SETROPTS CATDSNS is in effect, and either of the following is true:
    - The access request has been passed by a discrete or fully qualified generic profile.
    - The data set is a system temporary data set that is not protected by a profile in the DATASET class.

    RACF denies the request.

    RACF makes an exception and grants the request if the user is SPECIAL or has READ access to profile ICHUNCAT.*dataset-name* in the FACILITY class.
30. The postprocessing exit (ICHRCX02) can grant or deny the request.
31. For the DATASET class, if no profile is found and the SETROPTS PROTECTALL(FAILURES) option is in effect, RACF denies the request. If no profile is found and the SETROPTS PROTECTALL(WARNING) option is in effect, RACF grants the request.

Pictorial View of RACF Authorization Checking

For a complete description of the numbered steps described in the following figures, see "Authorizing Access to RACF-Protected Resources" on page 634.

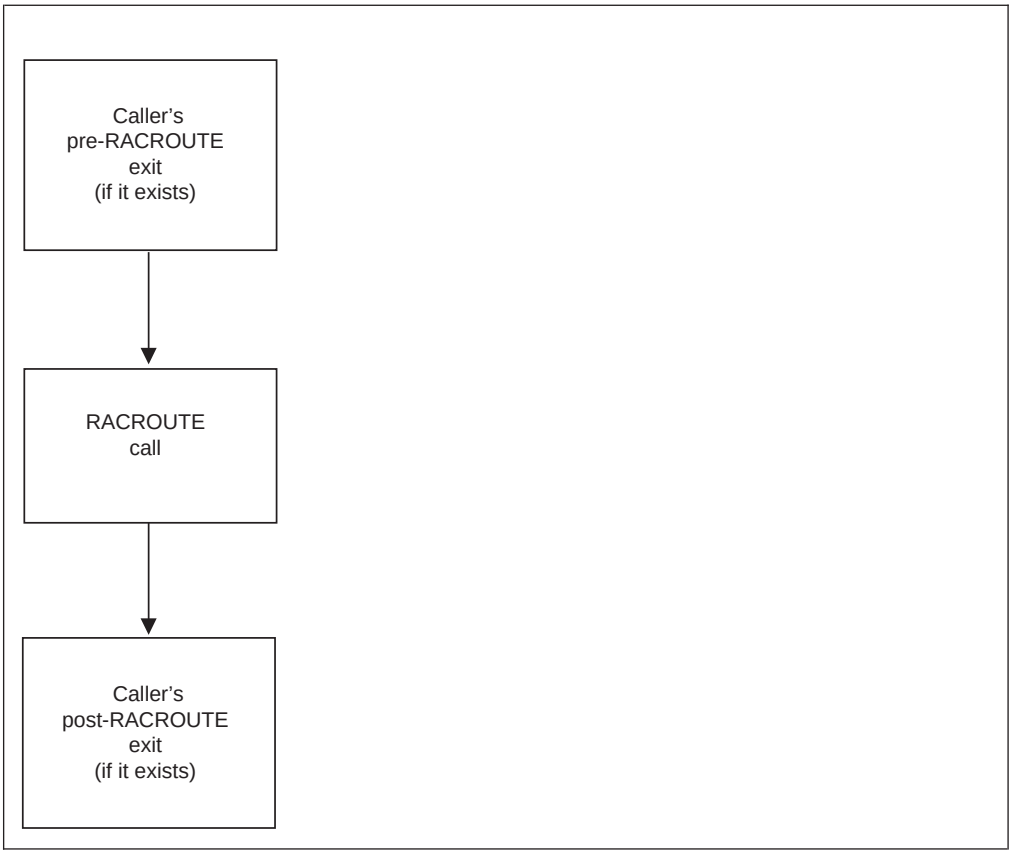


Figure 69. Process Flow of Callers of RACF for RACROUTE REQUEST=AUTH Requests

Note: For information about exits that affect RACROUTE calls, see the documentation for the calling product.

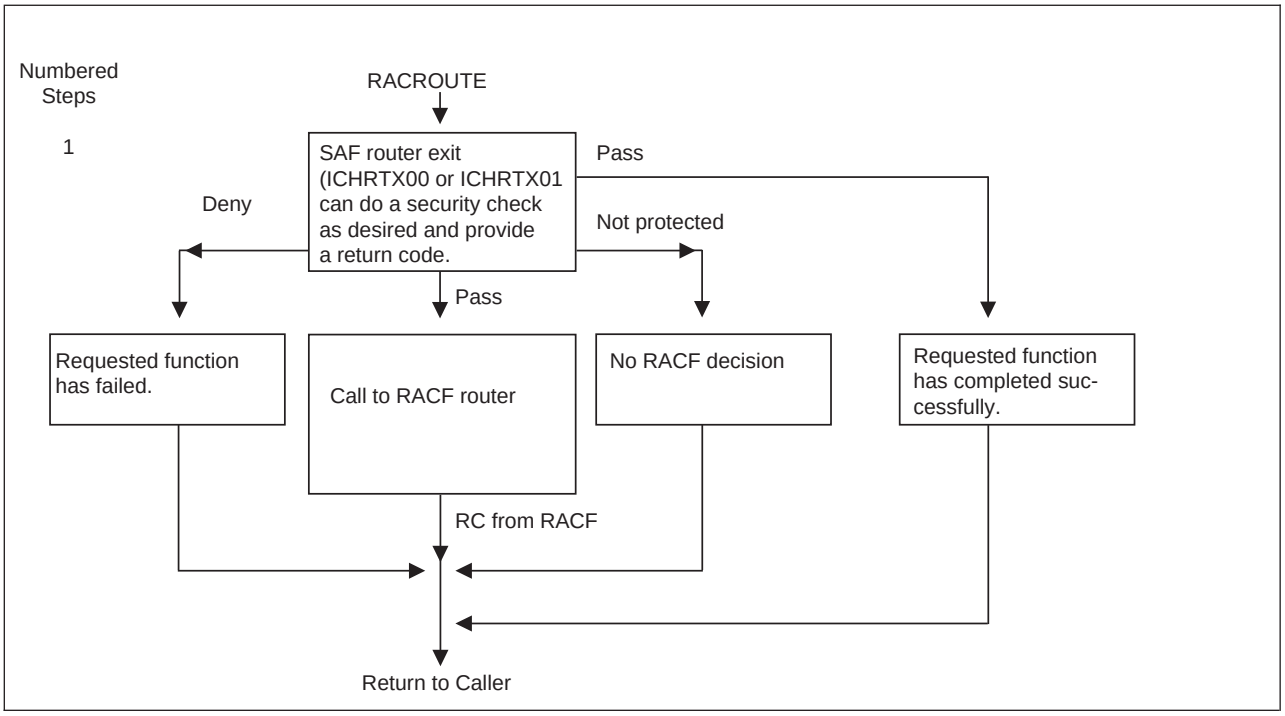


Figure 70. Process Flow of SAF Router for RACROUTE REQUEST=AUTH Requests

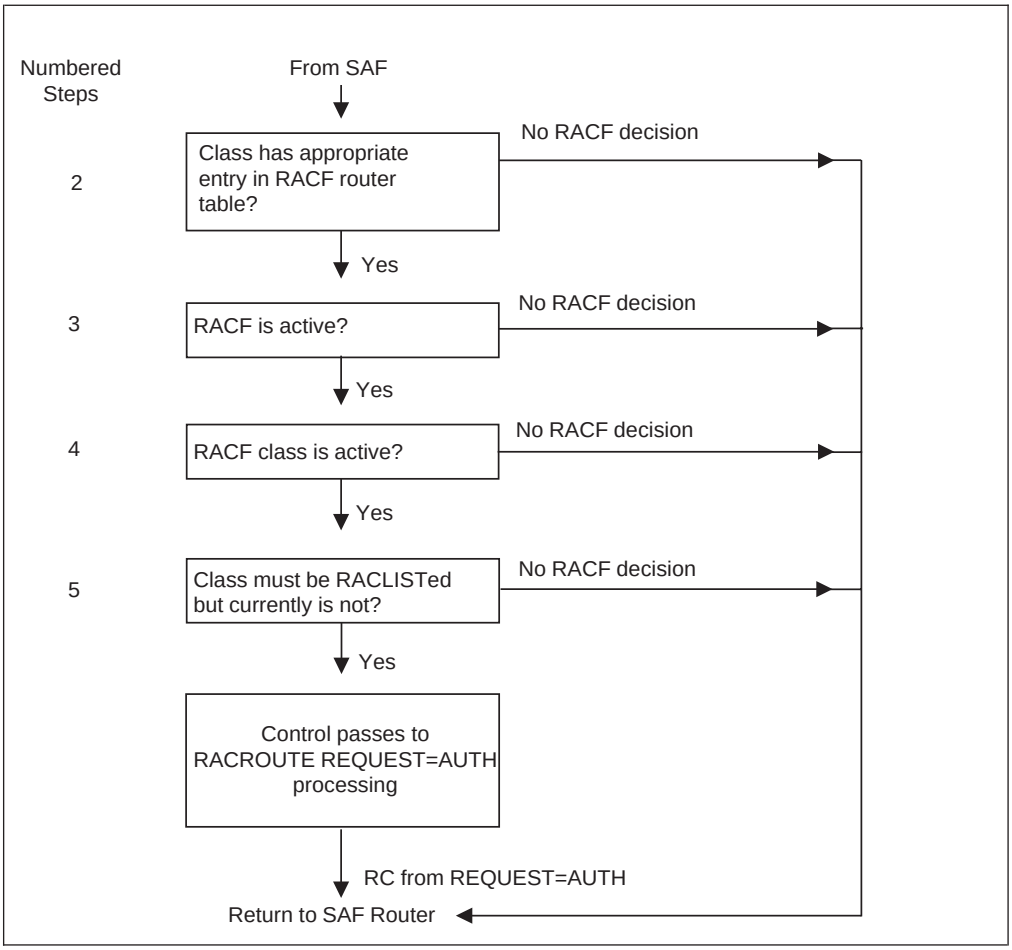


Figure 71. Process Flow of RACF Router

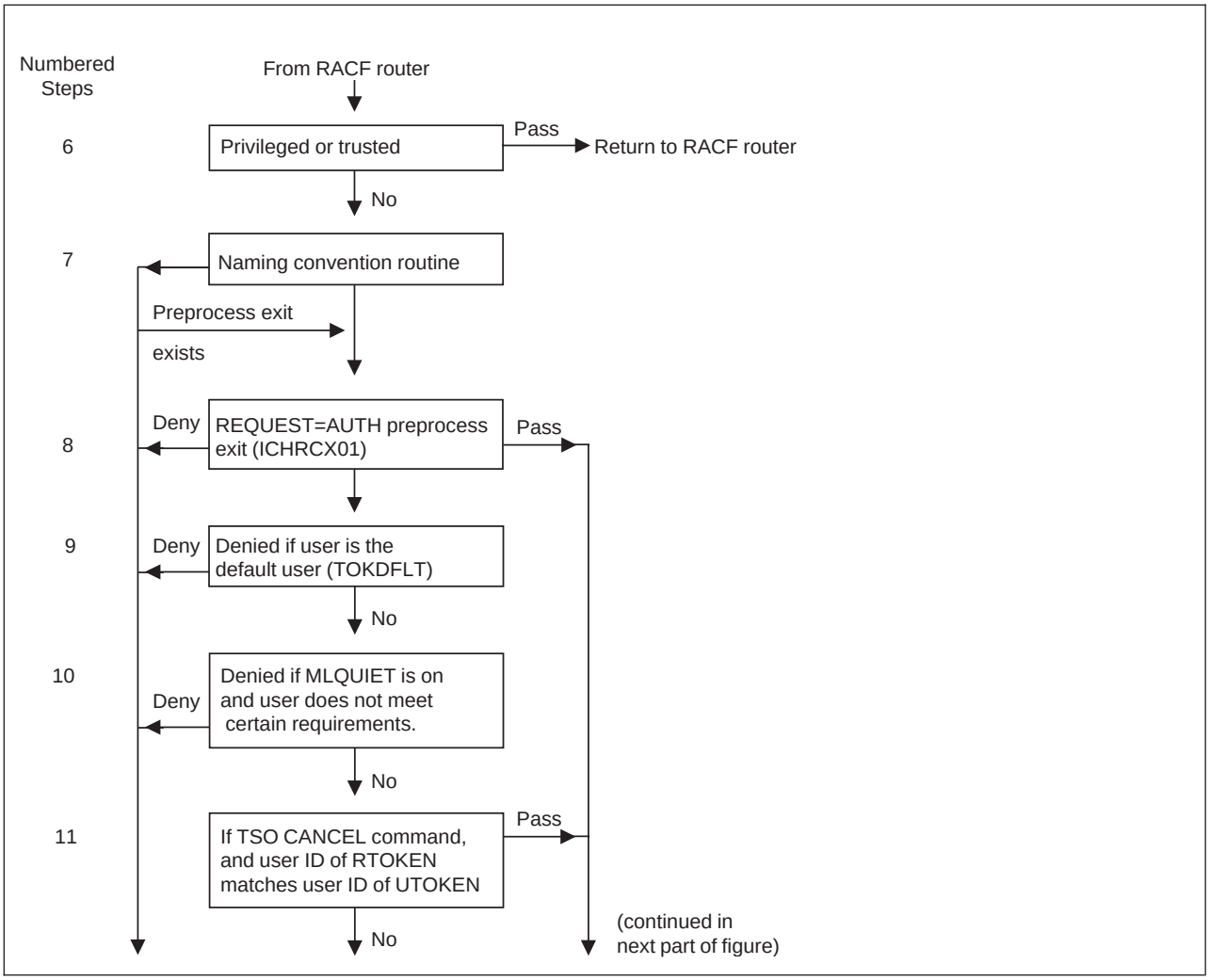


Figure 72. Process Flow of RACF Authorization Checking (Part 1 of 3)

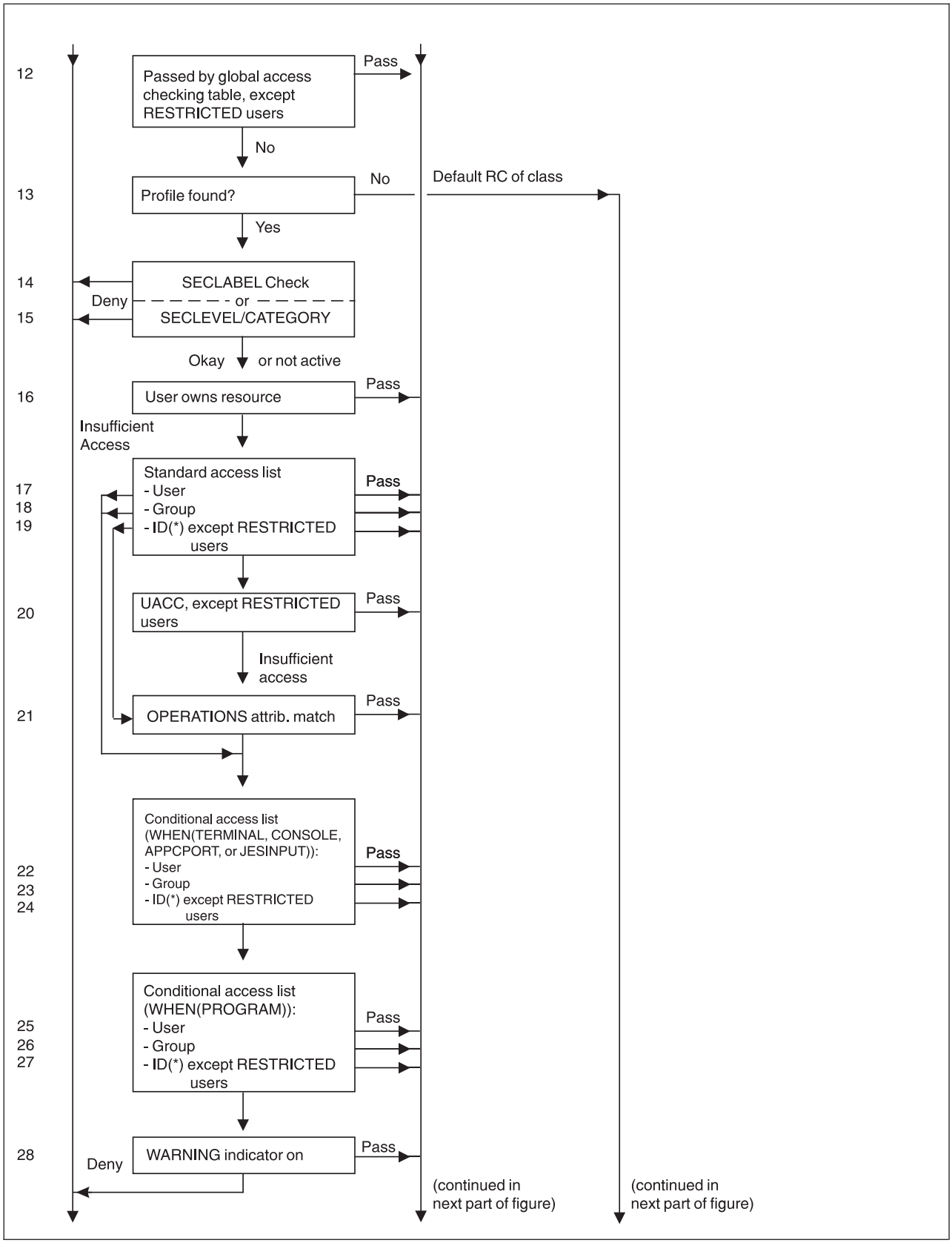


Figure 72. Process Flow of RACF Authorization Checking (Part 2 of 3)

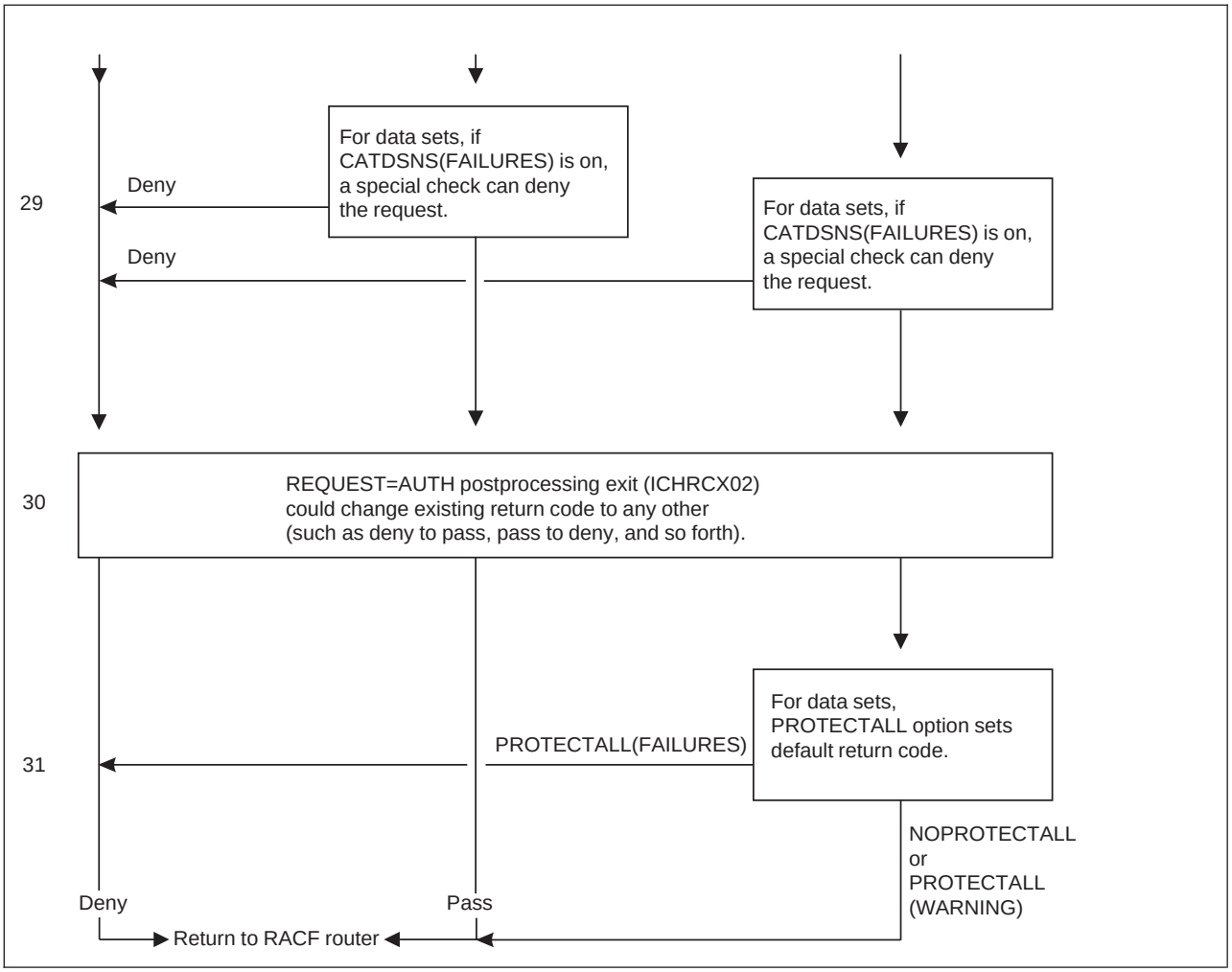


Figure 72. Process Flow of RACF Authorization Checking (Part 3 of 3)

Authorizing Access to RACF-Protected Terminals

When a RACF-defined user logs on to TSO or signs on to IMS or CICS using a terminal protected by a profile in the TERMINAL or GTERMINL class and the TERMINAL class is active, RACF performs authorization checking to verify that the user is permitted use of the terminal. RACF performs this authorization checking during REQUEST=VERIFY processing at the same time as it performs user identification and verification.

RACF performs terminal authorization checking in the following sequence:

1. If your installation has activated the SECLABEL class, RACF performs security label authorization checking. For a complete description, see “Security Label Authorization Checking” on page 647. If security label authorization checking succeeds, RACF authorization checking continues with the next step. To check what your system configuration should be, see “B1 Configuration Requirements” on page 12.
2. If the requesting user has at least READ access authority to the terminal, RACF processing continues at step 5. If the user’s access authority is NONE, RACF denies use of the terminal and stops terminal authorization checking.

3. If the requesting user's current connect group (or, if you activate list-of-groups checking, one of the user's other connect groups) has at least READ access authority to the terminal, RACF processing continues at step 5. If the group's access authority is NONE, RACF denies use of the terminal and stops terminal authorization checking.
4. If the profile has a universal access authority (UACC) of at least READ and your installation has not specified NOTERMUACC for the user's current connect group, RACF processing continues at step 5. Otherwise, RACF denies use of the terminal and stops terminal authorization checking.

   Note: For defined terminals, you can specify the universal access authority (UACC) with the RDEFINE or RALTER command. For undefined terminals, you can specify the universal access authority with the TERMUACC operand of the SETROPTS command.

   For more information, see "Limiting Specific Groups of Users to Specific Terminals" on page 237.
5. If your installation authorizes the use of the terminal on this particular day and time, RACF grants access to the terminal. (You can specify the terminal time and day-of-week restrictions with the RDEFINE and RALTER commands.) RACF also checks whether your installation has authorized the user to access the system on this particular day and time. (You can specify the user time and day-of-week restrictions with the ADDUSER and ALTUSER commands.)

Notes:

1. The REQUEST=AUTH and REQUEST=VERIFY preprocessing and postprocessing exit routines are available during terminal authorization checking.
2. Global access checking is not available during terminal authorization checking performed by REQUEST=VERIFY.
3. Profiles in the GTERMINL class are ignored unless SETROPTS RACLIST processing is in effect.

Authorizing Access to RACF-Protected Consoles, JES Input Devices, or APPC Partner LUs

When a RACF-defined user logs on to a JES or MCS console or submits a batch job from a JES input device, or submits an APPC request from a partner LU, RACF performs authorization checking to verify that the user is permitted use of the RACF-protected console or JES input device, or partner LU. RACF performs this authorization checking during REQUEST=VERIFY processing at the same time as it performs user identification and verification. For RACF to perform this authorization checking, your installation must activate the appropriate class, as follows:

- For JES or MCS consoles, activate the CONSOLE class.
- For JES input devices, activate the JESINPUT class.
- For APPC partner LUs, activate the APPCPORT class.

The resource must be protected by a profile in the appropriate class. If no profile exists for the resource, RACF fails the request.

Notes:

1. If the resource is a terminal, console, partner LU, or JES writer, RACF compares the security level of the user with the security level of the resource. If the resource has a higher security level than the user, RACF denies the request. For a terminal session, the security level that RACF uses for the user is the lower of the user's SECLEVEL and the terminal's SECLEVEL. Thus, if the

terminal has a SECLEVEL of 50 and the user has a SECLEVEL of 100, the user cannot access through that terminal any data that has a SECLEVEL of over 50.

2. RACF compares the list of security categories in the user's profile with the list of security categories in the resource's profile. If RACF finds any security category in the resource profile that is not in the user's profile, RACF denies the request. If RACF does not deny the request, RACF continues with authorization processing. If there are no categories in the resource profile, RACF continues with authorization processing.

The rest of this section uses the term *device authorization checking* to refer to the authorization checking done for any of the above resources.

RACF performs device authorization checking in the following sequence:

1. If your installation has activated the SECLABEL class, RACF performs security label authorization checking. For a complete description, see "Security Label Authorization Checking" on page 647. If security label authorization checking succeeds, RACF authorization checking continues with the next step. To check your system configuration, see "B1 Configuration Requirements" on page 12.
2. If the requesting user has at least READ access authority to the device, RACF grants the request with no further processing. If the user's access authority is NONE, RACF denies use of the device and stops device authorization checking. If the requesting user is not in the access list, device authorization checking continues with the next step.
3. If the requesting user's current connect group (or, if you activate list-of-groups checking, one of the user's other connect groups) has at least READ access authority to the device, RACF grants the request with no further processing. If the group's access authority is NONE, RACF denies use of the device and stops device authorization checking. If the group is not in the access list, device authorization checking continues with the next step.
4. If the profile has a universal access authority (UACC) of at least READ, RACF grants the request with no further processing. Otherwise, RACF denies use of the device and stops device authorization checking.

   Notes:

   a. The TERMUACC operand of the SETROPTS command has no effect on consoles or JES input devices.
   b. You cannot specify time or day-of-week restrictions for consoles or JES input devices. (You can specify user time and day-of-week restrictions with the ADDUSER and ALTUSER commands.)

Notes:

1. The REQUEST=AUTH and REQUEST=VERIFY preprocessing and postprocessing exit routines are available during device authorization checking.
2. Global access checking is not available during device authorization checking performed by REQUEST=VERIFY.

Authorization Checking for RACROUTE REQUEST=FASTAUTH Requests

Some resource managers, such as CICS, have high performance requirements. In order to do resource authorization checking with RACF, they use RACF facilities to load all of the profiles for a given class into the user's storage area or into a

common storage area called a data space. The resource managers can do a “fast” authorization check against profiles in the user’s storage, or profiles in the data space, or both.

See *z/OS SecureWay Security Server RACF System Programmer’s Guide* for information about processing RACLISTed profiles. See *z/OS SecureWay Security Server External Security Interface (RACROUTE) Macro Reference* for information on using RACROUTE REQUEST=LIST to place profiles in storage.

Fast authorization checking is different from normal authorization checking as follows:

- The privileged and trusted attributes are ignored.
- The global access checking table is not used.
- Security labels are not used.
- Security tokens are not used.
- If reverification is required for the transaction (IMS only), the user must also enter the SIGN ON password with the transaction request.
- No messages or SMF records are created. However, if the fast authorization processing determines that auditing is necessary, it indicates this to its caller. The caller can then issue a normal authorization request to cause SMF logging to be performed.

Note: For details on using RACF to provide security for CICS, see the *CICS RACF Security Guide*.

Authorizing Access to RACF-Protected Applications

You can control access to some applications (for example, IMS and CICS regions) by defining them to RACF as resources in the APPL class. When the user attempts to sign on the application, the application uses RACF to verify the user’s identity and his authority to use that application. RACF does an authorization check to determine the user’s authorization to the application.

- If there is a matching profile in the APPL class, RACF performs normal authorization checking as described in “Authorizing Access to RACF-Protected Resources” on page 634.
- If there is no matching profile in the APPL class, RACF allows the user to access the application.

Notes:

1. The REQUEST=AUTH and REQUEST=VERIFY preprocessing and postprocessing exit routines are available during application authorization checking.
2. Global access checking is not available during application authorization checking performed by REQUEST=VERIFY.
3. See “Chapter 14. RACF and Information Management System (IMS)” on page 425 for more information on how to protect IMS with RACF.
4. See *CICS RACF Security Guide* for information on how to protect CICS with RACF.

Security Label Authorization Checking

Note: This sequence of authorization checks beginsin one of the sequences in “Authorization Checking for RACF-Protected Resources” on page 633.

Debugging

When the SECLABEL class is active on your system, and a user or job requests access to a resource, RACF compares the security label of the user with the security label of the resource. To check what system configuration you should have, see “B1 Configuration Requirements” on page 12.

1. If the user requesting access does not have a security label and the resource does have a SECLABEL, RACF fails the request.
2. If the SETROPTS MLACTIVE(FAILURES) option is in effect and the resource does not have a security label associated with it, and the resource class is DATASET or another class that requires security labels as defined in the class descriptor table (CDT), RACF fails the request.

   If the SETROPTS COMPATMODE option is in effect, and the user requesting access does not have the correct security label, RACF checks to see if the user has access to a security label that could allow the requested access to the resource.

   - If the user has no access to any such security label, RACF fails the request.
   - If the user does have access to such a security label, RACF continues authorization checking and logs the request.

   If the SETROPTS MLACTIVE (FAILURES) option is in effect, users and jobs that do not have an associated security label cannot enter the system and no REQUEST=AUTH processing is ever done for them.
3. If the SETROPTS MLS(FAILURES) option is in effect, RACF makes one of the following tests, and fails the request if the test fails.

   If the SETROPTS COMPATMODE option is in effect, and the user requesting access does not have the correct security label, RACF checks to see if the user has access to a security label that could allow the requested access to the resource.

   - If the user has no access to any such security label, RACF fails the request.
   - If the user does have access to such a security label, RACF continues authorization checking and logs the request.
   - **Test for Read-Only Requests**

     If the request is only to read from the resource, the user’s current security label must be greater than or equal to the security label of the resource. This is true when *both* of the following are true:

     – The security level used to define the user’s current security label is equal to or higher than the security level used to define the security label of the resource.

     – All of the categories (if any) used to define the security label of the resource are in the user’s current security label.

     When the user’s current security label and the resource security label are related in this way, the user’s security label is said to *dominate* the resource security label.

     Note: The TSO LISTBC command and the OPEN macro for READ are read-only requests.
   - **Test for Read/Write Requests**

     If the request is to both read from and write to the resource, the user’s current security label must have the same definition as the security label of the resource. (That is, the security level and categories that define the one

security label must be the same as the security level and categories that define the other security label. The names of the security labels do not have to be the same.)

Note: The OPEN macro for WRITE is a read-write request.

- **Test for Write-Only Requests**

  If the request is to write only, the security label of the resource must be greater than or equal to the user's current security label. (Note that this is the opposite of read-only requests.)

  The most common write-only request is the TSO SEND command where the recipient of the message is at a higher security label than the sender.

  OS/390 does not support write-only requests for data sets or tape volumes. Therefore all such write requests are both read and write requests, and the security labels must be equal. Users can make write-only requests for DB2.

4. If the SETROPTS MLS(WARNING) option is in effect, RACF makes the same checks as in step 3 on page 648. If the access check fails on a read/write or write-only request because the user's current security label is greater than the security label of the resource, RACF issues a warning message and grants the request.
5. If the SETROPTS NOMLS option is in effect, RACF makes the following tests and fails the request if the test fails:
   - **Test for Read-Only and Read/Write Requests**

     The user's current security label must be greater than or equal to the security label of the resource. This is true when *both* the following are true:

     – The security level used to define the user's current security label is equal to or higher than the security level used to define the security label of the resource.

     – All of the categories (if any) used to define the security label of the resource are in the user's current security label.

   - **Test for Write-Only Requests**

     The user's current security label must be greater than or equal to the security label of the resource **or** the security label of the resource must be greater than or equal to the user's current security label. If this is true, then **either** of the following is true:

     – All of the categories (if any) used to define the security label of the resource are in the user's current security label.

     – All of the categories (if any) used to define the user's current security label are in the security label of the resource.

     When the user's current security label and the resource security label are not related in this way, they are said to be *disjoint*. Two security labels are disjoint when (1) the set of security categories that defines the first does not include the set of security categories that defines the second, and (2) the set of security categories that defines the second does not include the set of security categories that defines the first.

If the resource is a JES spool data set, RACF uses the security label in the token associated with the data set (specified on the RTOKEN parameter of the RACROUTE REQUEST=AUTH macro). Otherwise, RACF uses the security label kept in the resource profile protecting the resource.

Authorization Summary for SETROPTS MLS(FAILURES)

The following security label authorization rules apply when the SECLABEL class is active and SETROPTS MLS(FAILURES) is in effect:

- **Read Only:** The user's current security label must be greater than or equal to the security label of the resource.
- **Write Only:** The security label of the resource must be greater than or equal to the user's current security label.
- **Read/Write:** The user's current security label must have the same definition as the security label of the resource.

A user is not allowed to write to a resource that has a security label that is less than the user's current security label. This inability to "write down" is enforced to ensure that a user does not declassify data.

Table 61. SECLABEL Authorization When SECLABEL Class and SETR MLS(FAILURES) Are Active

Relationship between User's Current SECLABEL and Resource SECLABEL	R/O request	R/W request	W/O request
User is greater than resource	Pass	Fail	Fail
Resource is greater than user	Fail	Fail	Pass
User is equivalent to resource	Pass	Pass	Pass
Disjoint	Fail	Fail	Fail

Authorization Summary for SETROPTS NOMLS

The following security label authorization rules apply when the SECLABEL class is active and SETROPTS NOMLS is in effect:

- **Read Only:** The user's current security label must be greater than or equal to the security label of the resource.
- **Write Only:** One of the following statements must be true:
  - The security label of the resource must be greater than the user's current security label.
  - The user's current security label must be greater than the security label of the resource.
  - The user's current security label must have the same definition as the security label of the resource.

  In other words, the user's current security label and the security label of the resource cannot be disjoint.
- **Read/Write:** The user's current security label must be greater than or equal to the security label of the resource.

Table 62. SECLABEL Authorization When SECLABEL Class and SETR NOMLS Are Active

Relationship between User's Current SECLABEL and Resource SECLABEL	R/O request	R/W request	W/O request
User is greater than resource	Pass	Pass (*)	Pass (*)
Resource is greater than user	Fail	Fail	Pass
User is equivalent to resource	Pass	Pass	Pass
Disjoint	Fail	Fail	Fail

(*) For these items if SETROPTS MLS(WARNING) is active instead of NOMLS, a warning message (ICH408I) is issued to the security console.

Authorization Summary for SETROPTS MLACTIVE

Table 63 describes the results of security label authorization when the SECLABEL class is active and either the user's or resource's security label is missing. The results vary depending on the SETROPTS MLACTIVE setting and whether or not the resource class being checked requires security labels. The supplied class descriptor table (ICHRRCDX) specifies which resource classes require security labels. For a listing of the supplied class descriptor table (CDT) entries, see the *z/OS SecureWay Security Server RACF Macros and Interfaces*.

Table 63. Effects of MLACTIVE Settings on Security Label Authorization

Environment	Missing User SECLABEL (Resource SECLABEL is present)	Missing Resource SECLABEL (User SECLABEL is present)	Missing User and Resource SECLABELs
MLACTIVE(FAILURES) and resource class requires SECLABELs	Fail (*)	Fail	Fail (*)
MLACTIVE(FAILURES) and resource class requires SECLABELs	Fail	Pass and warning message sent to security console	Pass and warning message sent to security console
NOMLACTIVE and resource class requires SECLABELs	Fail	Pass	Pass
MLACTIVE(FAILURES) and resource class does not require SECLABELs	Fail (*)	Pass	Pass (*)
MLACTIVE(WARNING) and resource class does not require SECLABELs	Fail	Pass	Pass
NOMLACTIVE and resource class does not require SECLABELs	Fail	Pass	Pass

(*) For these items, the user has a missing SECLABEL and SETROPTS MLACTIVE(FAILURES) is in effect, so the user would not be allowed to log on to the system. If the user logged on before MLACTIVE(FAILURES) was activated, authorization requests are passed or failed according to the entries in the table.

Special Access Rule for SPECIAL Users

If SETROPTS MLACTIVE(FAILURES) and SETROPTS MLS(FAILURES) are active, a SPECIAL user who is logged on at the SYSHIGH SECLABEL is allowed to access resources that do not have a security label. RACF issues a warning message instead of failing the request. If these two SETROPTS options were turned on without proper preparation (assigning security labels to resources), this enables the security administrator to access resources on the system. To prevent violation of the no write-down rules, read-only access to resources is allowed with a warning message, but write access fails.

Relationships among SECLABEL, SETROPTS MLS(FAILURES), SETROPTS MLACTIVE(FAILURES) and SETROPTS MLQUIET

Table 64 shows the relationships of the security labels and the SETROPTS MLS, MLACTIVE(FAILURES) and MLQUIET options.

Table 64. Relationships among SECLABEL, SETROPTS MLS(FAILURES), SETROPTS MLACTIVE(FAILURES), and SETROPTS MLQUIET

SECLABEL	MLS (FAILURES)	MLACTIVE (FAILURES)	MLQUIET	Effect
Active	Off	Off	Off	RACF uses security labels and allows writing to a lower security label.
Inactive	Off	Off	Off	Security labels have no effect on authorization checking.
Active	On	Off	Off	RACF uses security labels and prevents writing to a lower security label (“no write down”).
Inactive	On	Off	Off	Security labels have no effect on authorization checking.
Active	Off	On	Off	Those resources required to have SECLABELS by the class descriptor table (CDT), the DATASET class, and users must have security labels.
Inactive	On	Off	Off	Security labels have no effect on authorization checking.
Active	On	On	Off	All resources must be labeled, RACF uses security labels, and RACF prevents writing to a lower security label.
Inactive	On	On	Off	Security labels have no effect on authorization checking.
Either	Either	Either	On	All attempts to access the system or resources fail (unless the attempt is made by the trusted computing base, a security administrator, or a console operator).

Problems with User ID Authentication

This section includes the following information:

- “When Logon or Job Initialization Processing Takes Place and Why”
- “Logon/Job Initialization Processing” on page 653

When Logon or Job Initialization Processing Takes Place and Why

When a user requests access to the system, the application controlling the user’s access can issue the RACROUTE macro with REQUEST=VERIFY or

REQUEST=VERIFYX specified (or the RACINIT macro). For ease of reference, this section calls any such a request a verify request, and the program issuing the request is called an application.

Some of the places verify requests occur are:
- When interactive users log on (through TSO)
- When batch jobs are submitted through JES
- When NJE jobs or SYSOUT are received
- When APPC/MVS allocation requests are received
- When CICS, IMS, or NetView/Access Services allow users to sign on
- When other APF-authorized applications allow users to access the system

Based on the specifications on the verify request, RACF determines whether the requesting user is authorized to enter the system.

- If the user is authorized to enter the system, RACF returns a “successful” return code (return code 0) to the application. The application then allows the request to complete.
- If the user is not authorized to enter the system, RACF returns an “unauthorized” return code (other than 0) to the application. In general, the application then fails the request.

Notes:

1. The REQUEST=VERIFY and REQUEST=VERIFYX preprocessing and postprocessing exit routines are available during verification processing.
2. RACF authorization checks can be requested by RACF or the application (for example, to determine if a user is authorized to use a particular terminal). REQUEST=AUTH preprocessing and postprocessing exits are available during this authorization processing.
3. SMF log records or messages can be generated. (Failures are always recorded. Successes can be recorded if the application requests it on the REQUEST=VERIFY request).

Logon/Job Initialization Processing

When users cannot log on (or jobs cannot be initiated) or started procedures fail, check the following:

- For all types of users and jobs, check for an authorization message that indicates the cause of the failure, such as:
  - User profile not defined
  - User ID revoked
  - Invalid or no password
  - Invalid group name
  - Invalid or no security label (depending on RACF options)
  - Attempt to change password when RACF is in read-only mode or when the RACF database is locked

  If the application’s message does not clearly indicate the source of the problem, check the RACF message. This message (ICH408I or ICH409I) may provide more information.

  Notes:

  1. You can find the message in one of the following places:
     - The user’s terminal
     - The job log
     - The security console
     - The system log

Also, equivalent information is in audit records generated by RACF. Some information might be in audit records generated by the caller of RACF.

2. For NJE jobs and SYSOUT, be aware that NODES profiles can cause the user ID, connect group, and security label to be translated to local values.
3. For NJE jobs, if password verification is required by the NODES profile used to verify the user ID, any password sent with the job must be the password associated with the user ID on the execution node.
4. If the 408I message indicates that access was denied because of a revoked user ID, you may want to resume that user ID. Check if the user ID is associated with the started procedure. If there was a user ID associated with the started procedure, this started procedure could not have begun successfully. After you resume the user ID, you must restart the started procedure or re-IPL.

- REQUEST=VERIFY processing might do some RACF authorization checks for the user. Also, the caller of RACF, or initial EXECs or procedures that are invoked automatically might require RACF authorization checking.

  See Table 65 to see which resource classes could be checked from various types of sessions.
- Check if an installation exit is causing the problem. Candidates include:
  – The SAF exits
  – Exits in the caller of RACF, such as JES or TSO
  – The REQUEST=VERIFY exits

Table 65. Resource Classes Checked for Logon/Job Initialization Requests

Type of Session	Classes That Might Be Checked
TSO logons	TERMINAL, SECLABEL, TSOPROC, ACCTNUM, PERFGRP, TSOAUTH, and (depending on the user's TSO logon procedure) DATASET or others
CICS signons	TERMINAL, SECLABEL, and APPL
IMS signons	TERMINAL, SECLABEL, and APPL
Operators logging on to MCS consoles	CONSOLE and SECLABEL
Batch jobs	JESINPUT, SECLABEL, JESJOBS, SURROGAT
Inbound NJE jobs	NODES, JESINPUT, SECLABEL, JESJOBS, SURROGAT
Inbound SYSOUT	NODES, JESINPUT, SECLABEL
RJE remote signons or logons	JESINPUT, SECLABEL, FACILITY (checks for existence of RJE.*userid* profile)
For NJE and RJE remote (commands)	CONSOLE, NODES, SECLABEL, OPERCMDS, FACILITY (for each command, a check is made against the NJE.*userid* or RJE.*userid* profile in the FACILITY class)
MOUNT (MVS operator requests that a DASD device be made active), system address space, and started procedures	Check the STARTED class or started procedures table (ICHRIN03) entry
APPC/MVS allocation requests	APPCPORT, APPCLU, APPCTP, APPCSERV, APPCSI, SECLABEL, APPL, DATASET

Appendix G. Notices

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

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

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

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs

and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

This product contains code licensed from RSA Data Security Incorporated.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States, or other countries, or both:

BookManager
CICS
CICS/ESA
CICSPlex
DATABASE 2

DB2
DB2 Universal Database
DFSMSdfp
DFSMSdss
DFSMS/MVS
DFSORT
Hiperbatch
IBM
IBMLink
IMS
IMS/ESA
Language Environment
Library Reader
MQSeries
MVS
MVS/ESA
OS/390
Print Services Facility
QMF
RACF
Redbooks
Resource Link
RMF
S/390
SecureWay
SP
System/390
SystemView
TalkLink
VM/ESA
VTAM
WebSphere
z/Architecture
z/OS
z/VM
zSeries

Lotus and Lotus Notes are trademarks of Lotus Development Corporation in the United States, other countries, or both.

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

Tivoli, TME, and NetView are trademarks of International Business Machines Corporation or Tivoli Systems, Inc. in the United States, other countries, or both.

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

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

RACF Glossary

This glossary defines technical terms and abbreviations used in RACF documentation. If you do not find the term you are looking for, refer to the index of the appropriate RACF manual or view *IBM Glossary of Computing Terms*, available from: http://www.ibm.com/ibm/terminology

Sequence of Entries

For purposes of clarity and consistency of style, this glossary arranges the entries alphabetically on a letter-by-letter basis, which means:

- Only the letters of the alphabet are used to determine sequence, and
- Special characters and spaces between words are ignored.

Organization of Entries

Each entry consists of:

- A single-word term,
- A multiple-word term,
- An abbreviation for a term, or
- An acronym for a term.

This entry is followed by a commentary, which includes one or more items (definitions or references) and is organized as follows:

1. An item number, if the commentary contains two or more items.
2. A usage label, indicating the area of application of the term, for example, “In programming,” or “In TCP/IP.” Absence of a usage label implies that the term is generally applicable to IBM, or to data processing.
3. A descriptive phrase, stating the basic meaning of the term. The descriptive phrase is assumed to be preceded by “the term is defined as...”. The part of speech being defined is indicated by the opening words of the descriptive phrase: “To ... ” indicates a verb, and “Pertaining to ... ” indicates a modifier. Any other wording indicates a noun or noun phrase.
4. Annotative sentences, providing additional or explanatory information.
5. References, pointing to other entries or items in the dictionary.

References

The following cross-references are used in this glossary:

- **Contrast with:** This refers to a term that has an opposed or substantively different meaning.
- **See:** This refers the reader to (a) a related term, (b) a term that is the expanded form of an abbreviation or acronym, or (c) a synonym or more preferred term.
- **Synonym for:** This indicates that the term has the same meaning as a preferred term, which is defined in its proper place in the glossary.
- **Synonymous with:** This is a reference from a defined term to all other terms that have the same meaning.
- **Obsolete term for:** This indicates that the term should not be used and refers the reader to the preferred term.

Selection of Terms

A term is the word or group of words being defined. In this glossary, the singular form of the noun and the infinitive form of the verb are the terms most often selected to be defined. If the term has an acronym or abbreviation, it is given in parentheses immediately following the term. The abbreviation’s definition serves as a pointer to the term it abbreviates, and the acronym’s definition serves as a pointer to the term it represents.

A

access. The ability to use a protected resource.

access authority. (1) The privileges granted to a particular user or group when accessing a protected resource (such as the ability to read or to update a data set). For resources protected by RACF profiles, the access authorities are NONE, EXECUTE, READ, UPDATE, CONTROL, and ALTER. These authorities are hierarchical, with READ also granting EXECUTE, UPDATE granting READ, and so forth. (2) RACF also has access authorities of READ, WRITE, and EXECUTE (or SEARCH) when dealing with files and directories in the HFS. Note that these authorities are not hierarchical, and HFS files are not protected by RACF profiles, although they do have access authorities.

access list. Synonym for *standard access list*. Contrast with *conditional access list*.

Glossary

ACEE. (accessor environment element) A control block that contains a description of the current user's security environment, including user ID, current connect group, user attributes, and group authorities. An ACEE is constructed during user identification and verification. See *ENVR object*.

ADAU. See *automatic direction of application updates*.

ADSP. See *automatic data set protection*.

ADSP attribute. A user attribute that establishes an environment in which all permanent DASD data sets created by the user are automatically defined to RACF and protected with a discrete profile. See *automatic data set protection*.

Advanced Program-to-Program Communication (APPC). A set of interprogram communication services that support cooperative transaction processing in an SNA network. APPC is the implementation, on a given system, of SNA's LU type 6.2. See *LU type 6.2* and *APPC/MVS*.

APF-authorized. A type of system authorization using the authorized program facility (APF) that allows an installation to identify system or user programs that can use sensitive system functions. To maintain system security and integrity, a program must be authorized by the APF before it can access restricted functions, such as supervisor calls (SVC) or SVC paths.

API. See *application programming interface*.

APPC. See *Advanced Program-to-Program Communication*.

APPC application. See *transaction program (TP)*.

APPC/MVS. The implementation of SNA's LU 6.2 and related communication services in the MVS base control program.

application programming interface (API). A software interface that enables applications to communicate with each other. An API is the set of programming language constructs or statements that can be coded in an application program to obtain the specific functions and services provided by an underlying operating system or service program.

application user identity. An alternate name by which a RACF user can be known to an application.

appropriate privileges. Describes which users can perform an action (such as execute a command, issue a syscall, and so forth) in a UNIX environment. Usually refers to having superuser authority or an appropriate subset of superuser authority.

attribute. See *user attribute* and *group-related user attribute*.

AUDIT request. The issuing of the RACROUTE macro with REQUEST=AUDIT specified. An AUDIT request is a general-purpose security request that a resource manager can use to audit.

AUDITOR attribute. A user attribute that allows the user to specify logging options on the RACF commands and list any profile (including its auditing options) using the RACF commands. Contrast with *group-AUDITOR attribute*.

AUTH request. The issuing of the RACROUTE macro with REQUEST=AUTH specified. The primary function of an AUTH request is to check a user's authorization to a RACF-protected resource or function. The AUTH request replaces the RACHECK function. See *authorization checking*.

authentication. (1) Verification of the identity of a user or the user's eligibility to access an object. (2) Verification that a message has not been altered or corrupted. (3) A process used to verify the user of an information system or protected resources. See also *password*.

authority. The right to access objects, resources, or functions. See *access authority, class authority,* and *group authority*.

authorization checking. The action of determining whether a user is permitted access to a protected resource. Authorization checking refers to the use of RACROUTE REQUEST=AUTH, RACROUTE REQUEST=FASTAUTH, or any of the RACF callable services unless otherwise stated. Note, however, that other RACF functions can also perform authorization checking as a part of their processing. For example, RACROUTE REQUEST=VERIFY can also check a user's authority to use a terminal or application.

automatic command direction. An RRSF function that enables RACF to automatically direct certain commands to one or more remote nodes after running the commands on the issuing node. Commands can be automatically directed based on who issued the command, the command name, or the profile class related to the command. Profiles in the RRSFDATA class control to which nodes commands are automatically directed. See *automatic direction of application updates, automatic password direction,* and *command direction.*

automatic data set protection (ADSP). A system function, enabled by the SETROPTS ADSP specification and the assignment of the ADSP attribute to a user with ADDUSER or ALTUSER, that causes all permanent data sets created by the user to be automatically defined to RACF with a discrete RACF profile.

automatic direction. See *automatic command direction*, *automatic password direction*, and *automatic direction of application updates*.

automatic direction of application updates. An RRSF function that automatically directs ICHEINTY and RACROUTE macros that update the RACF database to one or more remote systems. Profiles in the RRSFDATA class control which macros are automatically directed, and to which nodes. See *automatic command direction* and *automatic password direction*.

automatic password direction. An RRSF function that extends password synchronization and automatic command direction to cause RACF to automatically change the password for a user ID on one or more remote nodes after the password for that user ID is changed on the local node. Profiles in the RRSFDATA class control for which users and nodes passwords are automatically directed. See *password synchronization, automatic command direction*, and *automatic direction of application updates*.

automatic profile. A tape volume profile that RACF creates when a RACF-defined user protects a tape data set. When the last data set on the volume is deleted, RACF automatically deletes the tape volume profile. Contrast with *nonautomatic profile*.

B

backup data set. A data set in the backup RACF database. For each data set in the primary RACF database, an installation should define a corresponding backup data set. See *backup RACF database*.

backup RACF database. A RACF database that reflects the contents of the primary RACF database. Backup RACF databases may be designated in the data set name table (ICHRDSNT) or specified at IPL time. You can switch to a backup database without a re-IPL if the primary RACF database fails. See *primary RACF database*.

base segment. The portion of a RACF profile that contains the fundamental information about a user, group, or resource. The base segment contains information that is common to all applications that use the profile.

BER. This term represents the Basic Encoding Rules specified in ISO 8825 for encoding data units described in abstract syntax notation 1 (ASN.1). See also *DER*.

block update command (BLKUPD). A RACF diagnostic command used to examine or modify the content of individual physical records in a RACF data set.

C

cache structure. A coupling facility structure that contains data accessed by systems in a sysplex. For more information, see *z/OS SecureWay Security Server RACF System Programmer's Guide*.

callable service. In z/OS UNIX, a request by an active process for a service. Synonymous with *syscall*.

category. See *security category*.

CDMF. See *Commercial Data Masking Facility*.

CDT. See *class descriptor table*.

certificate. See *digital certificate.*

certificate authority. An organization that issues digital certificates. The certificate authority authenticates the certificate owner's identity and the services that the owner is authorized to use, issues new certificates, renews existing certificates, and revokes certificates belonging to users who are no longer authorized to use them.

certificate-authority certificate. A type of certificate managed by RACF. See *digital certificate*.

certificate name filter. A general resource profile created by the RACDCERT MAP command that maps multiple user IDs to a digital certificate in order to simplify administration of certificates, conserve storage space in the RACF database, maintain accountability, or maintain access control granularity.

CICS. See *Customer Information Control System*.

class. A collection of RACF-defined entities (users, groups, and resources) with similar characteristics. Classes are defined in the class descriptor table (CDT), except for the USER, GROUP, and DATASET classes.

class authority (CLAUTH). An attribute enabling a user to define RACF profiles in a class defined in the class descriptor table. A user can have class authorities to zero or more classes.

class descriptor table (CDT). A table consisting of an entry for each class except the USER, GROUP, and DATASET classes. The CDT contains the classes supplied by IBM and the installation-defined classes.

classification model 1. See *single-subsystem scope*.

classification model 2. See *multiple-subsystem scope*.

CLAUTH attribute. See *class authority*.

command direction. An RRSF function that allows a user to issue a command from one user ID and direct that command to run in the RACF address space on the same system or on a different RRSF node, using the same or a different user ID. Before a command can be directed from one user ID to another, a user ID association must be defined between them using the RACLINK command.

command prefix facility (CPF). An MVS facility that provides a registry for command prefixes. CPF ensures

that two or more subsystems do not have the same or overlapping command prefixes for MVS operator commands.

Commercial Data Masking Facility (CDMF). An encryption function that uses a weaker key (40 bit) of the Data Encryption Standard (DES) algorithm. RACF uses CDMF to mask the data portion of RRSF transaction processing message packets. CDMF is part of the IBM Common Cryptographic Architecture.

common programming interface (CPI). An evolving application programming interface (API), supplying functions to meet the growing demands from different application environments and to achieve openness as an industry standard for communications programming. CPI-C provides access to interprogram services such as sending and receiving data, synchronizing processing between programs, and notifying a partner of errors in the communication.

conditional access list. The portion of a resource profile that specifies the users and groups that may access the resource at a specified level when a specified condition is true. For example, with program access to data sets, the condition is that the user must be executing the program specified in the access list. Contrast with *standard access list*.

coordinator system. In a RACF data sharing group, the system on which the system operator or administrator enters a RACF command that is propagated throughout the group. Contrast with *peer system*.

coupling facility. The hardware element that provides high-speed caching, list processing, and locking functions in a sysplex.

CPF. See *command prefix facility*.

CPI-C. See *common programming interface*.

current connect group. The group specified by a user when logging on to the system, or the user's default group if the user did not specify a group when logging on. With SETROPTS NOGRPLIST in effect, RACF uses the user's authority and this group's authority during access checking. With SETR GRPLIST in effect, RACF includes the authority of the user's other groups, if any, but the user still has only one ″current connect group″. You can use the &RACGPID variable in members of GLOBAL profiles to refer to the user's current connect group.

current security label. The security label that RACF uses in RACF authorization checking if the SECLABEL class is active. For interactive users, this is the security label specified when the user logged on, or (if no security label was specified) the default security label in the user's user profile. For batch jobs, this is the security label specified in the SECLABEL operand of the JOB statement, or (if no security label was specified) the user's current security label in the user profile associated with the job.

Customer Information Control System (CICS). A program licensed by IBM that provides online transaction processing services and management for critical business applications. CICS runs on many platforms (from the desktop to the mainframe) and is used in various types of networks that range in size from a few terminals to many thousands of terminals. The CICS application programming interface (API) enables programmers to port applications among the hardware and software platforms on which CICS is available. Each product in the CICS family can interface with the other products in the CICS family, thus enabling interproduct communication.

D

DASDVOL authority. A preferred alternative to assigning the OPERATIONS or group-OPERATIONS attribute, DASDVOL authority allows you to authorize operations personnel to access only those volumes that they must maintain. Using DASDVOL authority is also more efficient for functions such as volume dumping, because only one authorization check for the volume needs to be issued, instead of individual requests for each data set on the volume. Note that modern data management software (such as DFSMSdss) does not require DASDVOL authority. Contrast with *OPERATIONS attribute*, and *group-OPERATIONS attribute*.

Data Lookaside Facility (DLF). A facility that processes DLF objects. A DLF object contains data from a single data set managed by Hiperbatch. The user (an application program) is connected to the DLF object, and the connected user can then access the data in the object through normal QSAM or VSAM macro instructions.

data security. The protection of data from intentional or unintentional unauthorized disclosure, modification, or destruction.

data security monitor (DSMON). A RACF auditing tool that produces reports enabling an installation to verify its basic system integrity and data security controls.

data set profile. A profile that provides RACF protection for one or more data sets. The information in the profile can include the data set profile name, profile owner, universal access authority, access list, and other data. See *discrete profile* and *generic profile*.

data sharing group, RACF. A collection of one or more instances of RACF in a sysplex that have been identified to XCF and assigned to the group defined for RACF sysplex data sharing. RACF joins group IRRXCF00 when enabled for sysplex communication.

data sharing mode. An operational RACF mode that is available when RACF is enabled for sysplex communication. Data sharing mode requires installation of coupling facility hardware.

DB2 administrative authority. A set of privileges, often covering a related set of objects, and often including privileges that are not explicit, have no name, and cannot be specifically granted. For example, the ability to terminate any utility job is included in the SYSOPR authority.

DB2 explicit privilege. A privilege that has a name, and is held as the result of an SQL GRANT statement.

DCE. See *Distributed Computing Environment*.

default group. The group specified in a user profile that provides a default current connect group for the user. See *current connect group*.

DEFINE request. The issuing of the RACROUTE macro with REQUEST=DEFINE specified or using a RACF command to add or delete a resource profile causes a DEFINE request. The DEFINE request replaces the RACDEF function.

delegation. The act of giving users or groups the necessary authority to perform RACF operations.

DER. This term represents the Distinguished Encoding Rules, which are a subset of the Basic Encoding Rules. See also *BER*.

digital certificate. A digital document that binds a public key to the identity of the certificate owner, thereby enabling the certificate owner to be authenticated. A certificate is issued by a certificate authority.

RACF can manage three types of digital certificates:

- **Certificate-authority certificate.** A certificate associated with a certificate authority and is used to verify signatures in other certificates.
- **Site certificate.** A certificate associated with a server, or network entity other than a user or certificate authority.
- **User certificate.** A certificate associated with a RACF user ID that is used to authenticate the user's identity, and may also be used to represent a server.

DIRAUTH request. The issuing of the RACROUTE macro with REQUEST=DIRAUTH specified. A DIRAUTH request works on behalf of the message-transmission managers to ensure that the receiver of a message meets security-label authorization requirements.

directed command. A RACF command that is issued from a user ID on an RRSF node. It runs in the RACF subsystem address space on the same or a different RRSF node under the authority of the same or a different user ID. A directed command is one that specifies AT or ONLYAT. See *command direction* and *automatic command direction*.

discrete profile. A resource profile that provides RACF protection for a single resource. Contrast with *generic profile* and *fully-qualified generic profile*.

discretionary access control. An access control environment in which the resource owner determines who can access the resource. Contrast with *mandatory access control*.

disjoint. Pertaining to security labels, when the set of security categories that defines the first does not include the set of security categories that defines the second, and the set of security categories that defines the second does not include the set of security categories that defines the first. This also means that the first does not dominate the second and the second does not dominate the first. See *dominate*.

Distributed Computing Environment (DCE). The Open Group specification (or a product derived from this specification) that assists in networking. DCE provides such functions as authentication, directory service (DS), and remote procedure call (RPC).

DLF object. When Data Lookaside Facility (DLF) is active, the first attempt to access a QSAM or VSAM data set defined to DLF creates a DLF object. A DLF object contains data from a single data set managed by Hiperbatch. The user (an application program) is connected to the DLF object, and the connected user can then access the data in the object through normal QSAM or VSAM macro instructions.

dominate. One security label dominates a second security label when the security level that defines the first is equal to or greater than the security level that defines the second, and the set of security categories that defines the first includes the set of security categories that defines the second.

DSMON. See *data security monitor*.

E

effective group identifier (effective GID). When the user connects to the system (for example, logs on to a TSO/E session), one group is selected as the user's current group. When a user becomes a z/OS UNIX user, the GID of the user's current group becomes the effective GID of the user's process. The user can access resources available to members of the user's effective GID. See *group identifier (GID)* and contrast with *real GID*.

effective user identifier (effective UID). When a user becomes a z/OS UNIX user, the UID from the user's RACF user profile becomes the effective UID of the user's process. The system uses the effective UID to determine if the user is a file owner. See *user identifier (UID)* and contrast with *real UID*.

entity. A user, group, or resource (for example, a DASD data set) that is defined to RACF.

ENVR object. A transportable form of the ACEE that can be used within a single system to create the original ACEE without accessing the RACF database. It can be used, with limits, elsewhere in a single sysplex to recreate the original ACEE without accessing the RACF database.

erase-on-scratch. The physical overwriting of data on a DASD data set when the data set is deleted (scratched).

EXTRACT request. The issuing of the RACROUTE macro with REQUEST=EXTRACT specified. An EXTRACT request retrieves or replaces certain specified fields from a RACF profile or encodes certain clear-text (readable) data. The EXTRACT request replaces the RACXTRT function.

F

failsoft processing. (1) Processing that occurs when no data sets in the primary RACF database are available (RACF is installed but inactive). RACF cannot make decisions to grant or deny access. The operator is prompted frequently to grant or deny access to data sets. The resource manager decides on the action for general resource classes with a return code of 4. (2) Failsoft processing can also occur as the result of RVARY INACTIVE (temporary failsoft) or as the result of a serious system error requiring a re-IPL (permanent failsoft).

FASTAUTH request. The issuing of the RACROUTE macro with REQUEST=FASTAUTH specified. The primary function of a FASTAUTH request is to check a user's authorization to a RACF-protected resource or function. A FASTAUTH request uses only in-storage profiles (brought into storage using RACF functions such as RACROUTE REQUEST=LIST) for faster performance than an AUTH request. The FASTAUTH request replaces the FRACHECK function. See *authorization checking*.

field-level access checking. The RACF facility by which a security administrator can control access to segments, other than the base segment, in a RACF profile and fields in those segments.

file permission bits. In z/OS UNIX, information about a file that is used, along with other information, to determine if a process has read, write, or execute/search permission to a file or directory. The bits are divided into three parts, which are owner, group, and other.

file security packet (FSP). In z/OS UNIX, a control block containing the security data (file's owner user identifier (UID), owner group identifier (GID), and the permission bits) associated with the file. This data is stored with the file in the file system.

file transfer protocol (FTP). In the Internet suite of TCP/IP-related protocols, an application-layer protocol that transfers bulk-data files between machines or hosts.

FMID. See *function modification identifier*.

FRACHECK request. RACROUTE REQUEST=FASTAUTH replaces the FRACHECK function. See *FASTAUTH request*.

FSP. See *file security packet*.

FTP. See *File Transfer Protocol*.

fully-qualified generic profile. A DATASET profile that was defined using the GENERIC operand and has a name that contains no generic characters. A fully-qualified generic profile protects only resources whose names exactly match the name of the profile. Contrast with *discrete profile* and *generic profile*.

function modification identifier (FMID). A 7-character identifier that is used in elements associated with z/OS and OS/390 to identify the release of the element.

G

GDG. See *generation data group*.

general resource. Any resource, other than an MVS data set, that is defined in the class descriptor table (CDT). General resources include DASD volumes, tape volumes, load modules, terminals, IMS and CICS transactions, and installation-defined resource classes.

general resource profile. A profile that provides RACF protection for one or more general resources. The information in the profile can include the general resource profile name, profile owner, universal access authority, access list, and other data.

general user. A user who has limited RACF privileges, such as logging on, accessing resources, and creating data sets. General users typically use and create RACF-protected resources, but have no authority to administer resources other than their own.

generation data group (GDG). A collection of data sets with the same base name, such as PAYROLL, that are kept in chronological order. Each data set in the GDG is called a generation data set, and has a name such as PAYROLL.G0001V00, PAYROLL.G0002V00, and so forth.

generic profile. A resource profile that can provide RACF protection for zero or more resources. The resources protected by a generic profile have similar names and identical security requirements, though with

RACFVARS, a generic profile can protect resources with dissimilar names, too. For example, a generic data set profile can protect one or more data sets. Contrast with *discrete profile*.

global access checking. The ability to allow an installation to establish an in-storage table of default values for authorization levels for selected resources. RACF refers to this table before performing normal RACROUTE REQUEST=AUTH processing and grants the request without performing an AUTH request if the requested access authority does not exceed the global value. RACF uses this table to process AUTH requests faster and with less overhead (no checking of access lists, no auditing) when you have resources for which you decide to grant access to all users, except those with restricted user IDs. If the requested access does not exceed the access granted by the table, RACF bypasses most of its normal AUTH processing. Global access checking can grant the user access to the resource, but it cannot deny access.

global resource serialization. A mechanism using ENQ with the SYSTEMS option (or, in some older programs, the RESERVE option) to serialize resources across multiple z/OS or OS/390 images. It is used by RACF to serialize access to its database and to in-storage tables and buffers.

globally RACLISTed profiles. In-storage profiles for RACF-defined resources that are created by RACROUTE REQUEST=LIST and that are anchored from an ACEE. Globally RACLISTed in-storage profiles are shared across a system, such as the way that in-storage profiles created by SETROPTS RACLIST are shared. Contrast with *locally RACLISTed profiles*.

group. A collection of RACF-defined users who can share access authorities for protected resources.

group-ADSP attribute. A group-related user attribute similar to the ADSP attribute for a user, but assigned by using the CONNECT command to restrict its effect to those cases where the user creates data sets with that group as the high level qualifier of the data set name (or as determined by the naming convention table or exit).

group-AUDITOR attribute. A group-related user attribute similar to the AUDITOR attribute for a user, but assigned by using the CONNECT command to restrict the user's authority to resources that are within the scope of the group. Contrast with *AUDITOR attribute*.

group authority. An authority specifying which functions a user can perform in a group. The group authorities are USE, CREATE, CONNECT, and JOIN.

group data set. A RACF-protected data set in which either the high-level qualifier of the data set name or the qualifier supplied by an installation-naming convention table or exit routine is a RACF group name.

group-GRPACC attribute. A group-related user attribute similar to the GRPACC attribute for a user, but assigned by using the CONNECT command to restrict its effect to the specific group. Contrast with *GRPACC attribute*.

group ID. Obsolete term for *group name*.

group identifier (GID). A number between 0 and 2 147 483 647 that identifies a group of users to z/OS UNIX. The GID is associated with a RACF group name when it is specified in the OMVS segment of the group profile. See *real GID.* Contrast with *effective group identifier (effective GID).*

group name. A string of 1–8 characters that identifies a group to RACF. The first character must be A through Z, # (X'7B'), $ (X'5B'), or @ (X'7C'). The rest can be A through Z, #, $, @, or 0 through 9.

group-OPERATIONS attribute. (1) A group-related user attribute similar to the OPERATIONS attribute for a user, but assigned by using the CONNECT command to restrict its effect to those resources that are within the scope of the group. (2) If a person needs to perform maintenance activities on DASD volumes, it is more efficient (for RACF processing) and better (for limiting the resources the person can access) to give the person authority to those volumes using the PERMIT command than to assign the person the OPERATIONS or group-OPERATIONS attribute. Contrast with *DASDVOL authority* and *OPERATIONS attribute*.

group profile. A profile that defines a group. The information in the profile includes the group name, profile owner, and users in the group.

grouping profile. A profile in a resource group class.

group-related user attribute. A user attribute, assigned at the group level, that enables the user to control the resource, group, and user profiles associated with the group and its subgroups. Group-related user attributes include group-SPECIAL attribute, group-AUDITOR attribute, and group-OPERATIONS attribute. Contrast with *user attribute*.

group-REVOKE attribute. Assigned through the CONNECT command that prevents the user from using that group as the current connect group. Also prevents RACF from considering that group during authorization checking.

group-SPECIAL attribute. A group-related user attribute similar to the SPECIAL user attribute, but it is assigned by the CONNECT command to restrict the user's authority to users, groups, and resources within the scope of the group. Within this scope, it gives the user full control over everything except auditing options. However, it does not give the user authority to change global RACF options that will affect processing outside the group's scope. Contrast with *SPECIAL attribute*.

GRPACC attribute. With this attribute, any group data sets that the user defines to RACF (through the ADSP attribute, the PROTECT operand on the DD statement, or the ADDSD command) are automatically made accessible to other users in the group at the UPDATE level of access authority if the user defining the profile is a member of the group. Contrast with *group-GRPACC attribute*.

H

HFS. See *hierarchical file system*.

hierarchical file system (HFS). The file system for z/OS UNIX that organizes data in a tree-like structure of directories.

I

ICB. See *inventory control block*.

ICHRIN03. See *started procedures table*.

interprocess communication facilities (IPC). IPC facilities are services that allow different processes to communicate. Message passing (using message queues), semaphore sets, and shared memory services are forms of interprocess communication facilities.

inventory control block (ICB). The first block in a RACF database. The ICB contains a general description of the database and, for the master primary data set, holds the RACF global options specified by SETROPTS.

IPC. See *interprocess communication facilities*.

issuer's distinguished name (IDN). The X.509 name that is associated with a certificate authority.

K

kernel. The part of z/OS UNIX that provides support for such services as UNIX I/O, process management, and general UNIX functionality.

kernel address space. The address space in which the z/OS UNIX kernel runs. See *kernel*.

key. In cryptography, a sequence of symbols that is used with a cryptographic algorithm for encrypting or decrypting data. See *private key* and *public key*.

key ring. A named collection of certificates for a specific user or server application used to determine the trustworthiness of a client or peer entity.

L

link pack area (LPA). An area of virtual storage containing reenterable routines from system libraries that are loaded at IPL time and can be used concurrently by all tasks in the system. The LPA presence in main storage saves loading time.

LIST request. The issuing of the RACROUTE macro with REQUEST=LIST specified. A LIST request builds in-storage profiles for a RACF general resource class. The LIST request replaces the RACLIST function.

list-of-groups checking. A RACF option (SETROPTS GRPLIST) that enables a user to access all resources available to all groups of which the user is a nonrevoked member, regardless of the user's current connect group. For any particular resource, RACF allows access based on the highest access among the groups in which the user is a member.

local logical unit (local LU). A logical unit that resides on the local system. Contrast with *partner logical unit (partner LU)*, or *remote logical unit (remote LU)*, which typically resides on a remote system. When both the local and partner LUs reside on the same system, the LU through which communication is initiated is the local LU, and the LU through which communication is received is the partner LU.

local mode. An RRSF node is operating in local mode when it has no RRSF logical node connection with any other RRSF node.

local transaction program (local TP). A transaction program that resides on the local system. Contrast with *partner transaction program (partner TP)*, which typically resides on a remote system.

locally RACLISTed profiles. In-storage profiles for RACF-defined resources that are created by RACROUTE REQUEST=LIST and that are anchored from an ACEE. Locally RACLISTed in-storage profiles are not shared across a system, the way that in-storage profiles created by SETROPTS RACLIST are shared. Contrast with *globally RACLISTed profiles*.

logging. The recording of audit data about specific events.

logical connection. See *RRSF logical node connection*.

logical unit (LU). A type of network accessible unit that enables users to gain access to network resources and communicate with each other.

logical unit type 6.2 (LU type 6.2). The SNA logical unit type that supports general communication between programs in a cooperative processing environment. Also, the SNA logical unit type on which CPI-C and APPC/MVS TP conversation services are built.

LPA. See *link pack area*.

LU. See *logical unit*.

LU type 6.2. See *logical unit type 6.2*.

M

MAC. See *mandatory access control*.

main system. The system on a multisystem RRSF node that is designated to receive most of the RRSF communications sent to the node.

managed user ID association. A user ID association in which one of the associated user IDs is a managing user ID, and the other is a managed user ID. The managing user ID can run allowed RACF commands under the authority of the managed user ID. The managed user ID cannot run commands under the authority of the managing user ID. A managed user ID association does not allow password synchronization between the associated user IDs. Contrast with *peer user ID association*.

mandatory access control (MAC). A means of restricting access to objects on the basis of the sensitivity (as represented by a label) of the information contained in the objects and the formal authorization (clearance) of subjects to access information of such sensitivity.

mask. A technique to provide protection against casual viewing of a password that has been defined or altered, when an encryption function is not available.

master primary data set. The first data set activated in the primary RACF database.

MCS. See *multiple console support*.

MCS console. A non-SNA device defined to MVS that is locally attached to an MVS system and is used to enter commands and receive messages.

member. A user belonging to a group.

member profile. A profile that defines a member and security level for that member.

member system. Any one of the MVS system images in a multisystem RRSF node.

modeling. See *profile modeling*.

multiple console support (MCS). The operator interface in an MVS system.

multiple-subsystem scope. A RACF classification model used in conjunction with the RACF/DB2 external security module to construct DB2 resource names. Default for the highest-level qualifier is the DB2 subsystem or group name.

multisystem node. See *multisystem RRSF node*.

multisystem RRSF node. An RRSF node consisting of multiple MVS system images that share the same RACF database. One of the systems is designated to be the main system, and it receives the unsolicited RRSF communications sent to the node.

multi-subsystem scope. A classification model used in conjunction with the RACF/DB2 external security module to construct DB2 classes with the subsystem ID as part of the class name. Contrast with *single-subsystem scope*.

MVS. (multiple virtual storage) The mainframe operating system that allows multiple users to work simultaneously using the full amount of virtual storage.

N

NCSC. National Computer Security Center. The part of the U.S. Department of Defense that determines defense and security criteria.

network-qualified name. An identifier for a partner LU in the form *netid.luname*, where *netid* is a 1–8 character network identifier and *luname* is a 1–8 character LU name.

node. See *RRSF node*.

nonautomatic profile. A tape volume profile that RACF creates when an RDEFINE command is issued or when tape data set protection is not active. A tape volume profile created in this manner is called a nonautomatic profile because RACF never deletes the profile except in response to the RDELETE command. Contrast with *automatic profile*.

non-data sharing mode. One of two normal modes of operation when RACF is enabled for sysplex communication and is the mode in which RACF communicates information using sysplex facilities to other instances of RACF, but does not make use of the coupling facility in doing so.

O

OpenExtensions VM. A feature of VM systems that provides a set of UNIX-based programming interfaces, such as shells and utilities, in support of selected POSIX and X/OPEN portability guide (XPG) standards.

OPERATIONS attribute. A user attribute that grants the equivalent of ALTER access to all data sets unless the user or one of the user's connect groups appears explicitly in the access list of a data set's profile. If a user needs to perform maintenance activities on DASD volumes, granting DASDVOL authority to those volumes using the PERMIT command is preferred over assigning the OPERATIONS or group-OPERATIONS attribute. Note that most modern DASD maintenance programs

do not require the OPERATIONS attribute. Contrast with *DASDVOL authority* and *group-OPERATIONS attribute*.

operator identification card (OIDCARD). A small card with a magnetic stripe encoded with unique characters and used to verify the identity of a terminal operator to RACF.

OS/390. A program licensed by IBM that not only includes and integrates functions previously provided by many IBM software products, including the MVS operating system, but also:

1. Is an open, secure operating system for the IBM S/390 family of enterprise servers
2. Complies with industry standards
3. Is Year 2000 ready and enabled for network computing and e-business
4. Supports technology advances in networking server capability, parallel processing, and object-oriented programming

OS/390 UNIX group identifier (GID). See *group identifier (GID)*.

OS/390 UNIX System Services (OS/390 UNIX). The set of functions provided by the shells, utilities, kernel, file system, debugger, Language Environment, and other elements of the OS/390 operating system that allows users to write and run application programs that conform to UNIX standards.

OS/390 UNIX user identifier (UID). See *user identifier (UID)*.

owner. The user or group that creates a profile, or is specified as the owner of a profile. The owner can modify, list, or delete the profile.

P

PADS. See *program access to data sets (PADS)*.

partner logical unit (partner LU). A logical unit that typically resides on a remote system. Often synonymous with *remote logical unit (remote LU)*. Contrast with *local logical unit (local LU)*, which resides on the local system. When both the local and partner LUs reside on the same system, the LU through which communication is initiated is the local LU, and the LU through which communication is received is the partner LU.

partner transaction program (partner TP). A transaction program that resides on a remote system. Contrast with *local transaction program (local TP)*, which typically resides on the local system.

PassTicket. An alternative to the RACF password that permits workstations and client machines to communicate with the host. It allows a user to gain access to the host system without sending the RACF password across the network.

password. A string of characters known to a user who must specify it to gain full or limited access to a system and to the data stored within it. RACF uses a password to verify the identity of the user.

password synchronization. An option that can be specified when a peer user ID association is defined between two user IDs. If password synchronization is specified for a user ID association, then whenever the password for one of the associated user IDs is changed, the password for the other user ID is automatically changed to the newly defined password. See *automatic password direction*.

peer system. In a RACF data sharing group, any system to which RACF propagates a command entered by the system operator or administrator. Contrast with *coordinator system*.

peer user ID association. A user ID association that allows either user ID to run allowed RACF commands under the authority of the other user ID using command direction. A peer user ID association can also establish password synchronization between the associated user IDs. Contrast with *managed user ID association*.

permission bits. In z/OS UNIX, part of security controls for directories and files stored in the hierarchical file system (HFS). Used to grant read, write, search (just directories), or execute (just files) access to owner, file or directory owning group, or all others.

persistent verification (PV). A VTAM security option for conversation-level security between two logical units (LUs) that provides a way of reducing the number of password transmissions by eliminating the need to provide a user ID and password on each attach (allocate) during multiple conversations between a user and a partner LU. The user is verified during the signon process and remains verified until the user has been signed off the partner LU.

POSIT. A number specified for each class in the class descriptor table that identifies a set of flags that control RACF processing options. See the description of the POSIT keyword in the *z/OS SecureWay Security Server RACF Macros and Interfaces*.

POSIX. (Portable Operating System Interface For Computer Environments) An IEEE standard for computer operating systems.

primary data set. A data set in the primary RACF database. See *master primary data set*.

primary RACF database. The RACF database designated in the data set name table (ICHRDSNT), or specified at IPL time, that contains the RACF profiles

used for authorization checking. The primary RACF database may consist of as many as 90 data sets. See *backup RACF database*.

private key. In public key cryptography, a key that is known only to its owner. Contrast with *public key*.

problem state. A state during which a processing unit cannot execute input/output and other privileged instructions. Contrast with *supervisor state*.

process. In z/OS UNIX, a function created by a **fork()** request. See *task*.

profile. Data that describes the significant characteristics of a user, a group of users, or one or more computer resources. A profile contains a base segment, and optionally, a number of other segments. See *data set profile, discrete profile, general resource profile, generic profile, group profile,* and *user profile*.

profile list. A list of profiles indexed by class (for general resources) or by the high-level qualifier (for data set profiles) and built in storage by the RACF routines.

profile modeling. The ability for a user or an installation to copy information (such as universal access authority or access lists) from an existing resource profile when defining a new resource profile. This might occur automatically when using ADDSD based on the MODEL specification in a USER or group PROFILE, or manually with the FROM keyword of the ADDSD and RDEFINE commands, or with keywords on RACROUTE REQUEST=DEFINE.

program access to data sets (PADS). A RACF function that enables an authorized user or group of users to access one or more data sets at a specified access authority only while running a specified RACF-controlled program. See *program control*.

program control. A RACF function that enables an installation to control who can run RACF-controlled programs. See *program access to data sets*.

protected resource. A resource defined to RACF for the purpose of controlling access to the resource. Some of the resources that can be protected by RACF are DASD volumes, tape volumes, load modules, terminals, IMS and CICS transactions, and installation-defined resource classes.

protected user ID. A user ID that cannot enter the system by any means that requires a password, and cannot be revoked by invalid password attempts. Assigning a protected user ID to z/OS UNIX, a UNIX daemon, or another important started task or subsystem assures that the ID cannot be used for other purposes, and that functions will not fail because the ID has been revoked.

public key. In public key cryptography, a key that is made available to everyone. Contrast with *private key*.

public key cryptography. Cryptography in which public keys and private keys are used for encryption and decryption. One party uses a common public key and the other party uses secret private key. The keys are complementary in that if one is used to encrypt data, the other can be used to decrypt it.

PV. See *persistent verification*.

R

RACDEF request. The DEFINE function replaces the RACDEF function. See *DEFINE request*.

RACF. See *Resource Access Control Facility*.

RACF/DB2 external security module. A RACF exit point that receives control from the DB2 access control authorization exit point (DSNX@XAC) to handle DB2 authorization checks.

RACF database. The repository for the security information that RACF maintains.

RACF data set. One of the data sets comprising the RACF database.

RACF-indicated. Pertaining to a data set for which the RACF indicator is set on. If a data set is RACF-indicated, a user can access the data set only if a RACF profile or an entry in the global access checking table exists for that data set. On a system without RACF, a user cannot access a RACF-indicated data set until the indicator is turned off. For VSAM data sets, the indicator is in the catalog entry. For non-VSAM data sets, the indicator is in the data set control block (DSCB). For data sets on tape, the indicator is in the RACF tape volume profile of the volume that contains the data set.

RACF manager. The routines within RACF that provide access to the RACF database. Contrast with *RACF storage manager*.

RACF-protected. Pertaining to a resource that has either a discrete profile or an applicable generic profile. A data set that is RACF-protected by a discrete profile must also be RACF-indicated.

RACF remote sharing facility (RRSF). RACF services that function within the RACF subsystem address space to provide network capabilities to RACF.

RACF remove ID utility. A RACF utility that identifies references to user IDs and group names in the RACF database. The utility can be used to find references to residual user IDs and group names or specified user IDs and group names. The output from this utility is a set of RACF commands that can be used to remove the references from the RACF database after review and possible modification. See *residual user ID*.

RACF report writer. A RACF function that produces reports on system use and resource use from information found in the RACF SMF records. However, the preferred method for producing RACF SMF reports is the RACF SMF data unload utility (IRRADU00).

RACF segment. Obsolete term for *base segment*.

RACF SMF data unload utility (IRRADU00). A RACF utility that enables installations to create a sequential file from the security-relevant audit data. The sequential file can be viewed directly, used as input for installation-written programs, and manipulated with sort/merge utilities. It can also be uploaded to a database manager (such as DB2) to process complex inquiries and create installation-tailored reports. See *SMF records*.

RACF storage manager. Manages the allocation of storage for the RACF programs running on a system.

RACHECK request. The AUTH request replaces the RACHECK function. See *AUTH request*.

RACINIT request. The RACINIT request replaces the RACINIT function. See *VERIFY request*.

RACLIST request. The LIST request replaces the RACLIST function. See *LIST request*.

RACLISTed profiles. See *locally RACLISTed profiles* and *globally RACLISTed profiles*.

RACROUTE macro. An assembler macro that provides a means of calling RACF to provide security functions, including the *AUDIT request, AUTH request, DEFINE request, DIRAUTH request, EXTRACT request, FASTAUTH request, LIST request, SIGNON request, STAT request, TOKENBLD request, TOKENMAP request, TOKENXTR request, VERIFY request,* and *VERIFYX request*.

RACSTAT request. The STAT request replaces the RACSTAT function. See *STAT request*.

RACXTRT request. The EXTRACT request replaces the RACXTRT function. See *EXTRACT request*.

RBA. See *relative byte address*.

read-only mode. A recovery mode of operation when RACF is enabled for sysplex communication. Read-only mode does not allow updates to be made to the RACF database except for statistics generated during logon and job initiation.

real GID. The attribute of a process that, at the time of process creation, identifies the group of the user who created the process. See *group identifier (GID)*. Contrast with *effective group identifier (effective GID)*.

real UID. The attribute of a process that, at the time of process creation, identifies the user who created the process. See *user identifier (UID)*. Contrast with *effective user identifier (effective UID)*.

relative byte address (RBA). The address in the RACF database.

remote logical unit (remote LU). A logical unit that resides on a remote system. Often synonymous with *partner logical unit (partner LU)*. Contrast with *local logical unit (local LU)*, which typically resides on the local system.

residual authority. References in the RACF database to group names and user IDs that have been deleted.

residual group name. References in the RACF database to a group name that has been deleted.

residual user ID. References in the RACF database to a user ID that has been deleted.

Resource Access Control Facility (RACF). A program (licensed by IBM) that provides access control by identifying and verifying the users to the system, authorizing access to protected resources, logging detected unauthorized attempts to enter the system, logging unauthorized attempts to enter the system, and logging detected accesses to protected resources. RACF is included in the SecureWay Security Server and is also available as a separate program for the MVS and VM environments.

resource group profile. A general resource profile in a resource grouping class. A resource group profile can provide RACF protection for one or more resources with *unlike* names. See *resource grouping class*.

resource grouping class. A RACF class in which resource group profiles can be defined. A resource grouping class is related to another class, sometimes called a *member class*. For example, the resource grouping class GTERMINL is related to the class TERMINAL. See *resource group profile*.

resource profile. A profile that provides RACF protection for one or more resources. USER, GROUP, and CONNECT profiles are not resource profiles. The information in a resource profile can include the profile name, profile owner, universal access authority, access list, and other data. Resource profiles can be discrete profiles or generic profiles. See *discrete profile* and *generic profile*.

RESTRICTED attribute. A user attribute that can be assigned to a shared user ID, such as PUBLIC or ANONYMOS, or a user ID used with a certificate name filter, to prevent the user ID from being used to access protected resources it is not specifically authorized to access. Restricted users cannot gain access to protected resources through global access checking, UACC, or an `ID(*)` entry on the access list.

REVOKE attribute. A user attribute that prevents a RACF-defined user from entering the system.

role. In Tivoli products, a functional grouping of user authorizations. A ROLE profile represents a role and identifies the authorizations associated with that role.

RRSF. See *RACF remote sharing facility*.

RRSF logical node connection. Two RRSF nodes are logically connected when they are properly configured to communicate through APPC/MVS, and they have each been configured by the TARGET command to have an OPERATIVE connection to the other.

RRSF network. Two or more RRSF nodes that have established RRSF logical node connections to each other.

RRSF node. An MVS system image or a group of MVS system images sharing a RACF database, which has been defined as an RRSF node, single-system RRSF node, or multisystem RRSF node to RACF by a TARGET command. See *RRSF logical node connection*.

RTOKEN. The RACF resource security token. An RTOKEN is an encapsulation or representation of the security characteristics of a resource. Resource managers, for example JES, can assign RTOKENs to the resources they manage; for example, JES spool files. See *UTOKEN* and *STOKEN*.

S

SAF. See *System Authorization Facility*.

secured signon. A RACF function providing an alternative to the RACF password and also providing enhanced security across a network.

SecureWay Security Server. A licensed feature of z/OS that is comprised of Resource Access Control Facility (RACF), DCE Security Server, Lightweight Directory Access Protocol (LDAP) Server, z/OS Firewall Technologies, Open Cryptographic Enhanced Plug-ins (OCEP), and Network Authentication Service.

security. See *data security*.

security category. An installation-defined name corresponding to a department or area within an organization whose members have similar security requirements.

security classification. The use of security categories, a security level, or both, to impose additional access controls on sensitive resources. An alternative way to provide security classifications is to use security labels.

security label. An installation-defined name that corresponds to a specific RACF security level with a set of zero or more security categories. This is equivalent to the NCSC term *sensitivity label*.

security level. An installation-defined name that corresponds to a numerical security level; the higher the number, the higher the security level.

Security Server. See *SecureWay Security Server*.

security token. A collection of identifying and security information that represents data to be accessed, a user, or a job. This contains a user ID, group name, security label, node of origin, and other information.

segment. A portion of a profile. The format of each segment is defined by a template.

SETROPTS RACLISTed profiles. See *globally RACLISTed profiles*.

SFS. See *Shared File System*.

shared file system (SFS). On VM/ESA, a part of CMS that lets users organize their files into groups known as directories and selectively share those files and directories with other users.

signed-on-from list. A list of user entries identifying those users who have been signed on from a partner LU to a local LU and is associated with persistent verification.

SIGNON request. The issuing of the RACROUTE macro with REQUEST=SIGNON specified. A SIGNON request is used to manage the signed-on-from lists associated with persistent verification.

single-subsystem scope. A classification model used in conjunction with the RACF/DB2 external security module to construct DB2 classes with the subsystem ID as part of the class name. Contrast with *multi-subsystem scope*.

single-system node. See *single-system RRSF node*.

single-system RRSF node. An RRSF node consisting of one MVS system image.

site certificate. A type of certificate managed by RACF. See *digital certificate*.

SMF. See *System Management Facility*.

SMF data unload utility. See *RACF SMF data unload utility*.

SMF records. (1) Records and system or job-related information collected by the System Management Facility (SMF) and used in billing users, reporting reliability, analyzing the configuration, scheduling jobs, summarizing direct access volume activity, evaluating data set activity, profiling system resource use, and maintaining system security. (2) Variable-length process or status records from the SMF data set that are written

to the SMF log data set. These records vary in layout based on the type of system information they contain. See *RACF SMF data unload utility*.

SMS. See *Storage Management Subsystem*.

SNA. See *System Network Architecture (SNA)*.

source user ID. The source half of a source user ID and target user ID pair that has an established user ID association between them. For command direction the source user ID is the user ID that issued the command that is being directed. For password synchronization the source user ID is the user ID whose password changed, causing a change to the password of the target user ID. Contrast with *target user ID*.

SPECIAL attribute. A user attribute that gives the user full control over all of the RACF profiles in the RACF database and allows the user to issue all RACF commands, except for commands and operands related to auditing. Contrast with *group-SPECIAL attribute*.

split database. A RACF database that has been divided among multiple data sets.

standard access list. The portion of a resource profile that specifies the users and groups that may access the resource and the level of access granted to each. Synonymous with *access list*. Contrast with *conditional access list*.

started procedures table (ICHRIN03). Associates the names of started procedures with specific RACF user IDs and group names. It can also contain a generic entry that assigns a user ID or group name to any started task that does not have a matching entry in the table. However, it is recommended that you use the STARTED class for most cases rather than the started procedures table.

STAT request. The issuing of the RACROUTE macro with REQUEST=STAT specified. A STAT request determines if RACF is active and (optionally) if a given resource class is defined to RACF and active. The STAT request replaces the RACSTAT function.

STOKEN. A UTOKEN associated with a user who has submitted work. See *UTOKEN* and *RTOKEN*.

Storage Management Subsystem (SMS). A DFSMS facility used to automate and centralize storage management by providing the storage administrator with control over data class, storage class, management class, storage group, and automatic class selection routine definitions.

structure. See *cache structure*.

stub. (1) A function that connects with the specified library, but remains outside the specified library. (2) A protocol extension procedure.

subject's distinguished name (SDN). The X.509 name in a digital certificate that is associated with the name of the subject.

superuser. In z/OS UNIX, a system user who operates with the special privileges needed to perform a specified administrative task.

superuser authority. In z/OS UNIX, the unrestricted authority to access and modify any part of the operating system, usually associated with the user who manages the system.

supervisor. The part of a control program that coordinates the use of resources and maintains the flow of processing unit operations. Synonym for *supervisory routine*.

supervisor state. A state during which a processing unit can execute input/output and other privileged instructions. Contrast with *problem state*.

supervisory routine. A routine, usually part of an operating system, that controls the execution of other routines and regulates the flow of work in a data processing system. Synonymous with *supervisor*.

syscall. See *callable service*.

sysplex (system complex). Multiple systems communicating and cooperating with each other through multisystem hardware elements and software services to process the installation's workloads.

sysplex communication. An optional RACF function that allows the system to use XCF services and communicate with other systems that are also enabled for sysplex communication.

system authorization facility (SAF). An interface defined by MVS that enables programs to use system authorization services in order to control access to resources, such as data sets and MVS commands. SAF either processes security authorization requests directly or works with RACF, or other security product, to process them.

system call. In z/OS UNIX, a synonym for *callable service*.

system complex. See *sysplex*.

System Management Facility (SMF). The part of the operating system that collects and records system and job-related information used in billing users, reporting reliability, analyzing the configuration, scheduling jobs, summarizing direct access volume activity, evaluating data set activity, profiling system resource use, and maintaining system security. The information is recorded in the SMF log data set.

Systems Network Architecture (SNA). The IBM architecture that defines the logical structure, formats,

protocols, and operational sequences for transmitting information units through, and controlling the configuration and operation of, networks. The layered structure of SNA allows the ultimate origins and destinations of information, that is, the users, to be independent of and unaffected by the specific SNA network services and facilities used for information exchange.

T

tape volume set. The collection of tape volumes on which a multivolume data set resides. A volume set is represented in one RACF profile.

tape volume table of contents (TVTOC). Information about a tape data set that RACF stores in the tape volume profile for the volume on which the data set resides. The TVTOC includes the data set name, data set sequence number, creation date, and an indicator as to whether a discrete tape data set profile exists.

target node. An RRSF node that a given RRSF node is logically connected to, as a result of a TARGET command. See *local node* and *remote node*.

target user ID. The target half of a source user ID and target user ID pair that has an established user ID association between them. For command direction, the target user ID is the user ID specified on the AT or ONLYAT keyword, and is the user ID under whose authority the command is run on the specified node. For password synchronization, the target user ID is the user ID whose password RACF automatically updates when the password for the source user ID is changed. Contrast with *source user ID.*

task. A basic unit of work to be performed or a process and the procedures that run the process.

template. Contains mappings of the profiles on the RACF database.

TOKENBLD request. The issuing of the RACROUTE macro with REQUEST=TOKENBLD specified. A TOKENBLD request builds a UTOKEN.

TOKENMAP request. The issuing of the RACROUTE macro with REQUEST=TOKENMAP specified. A TOKENMAP request maps a token in either internal or external format, allowing a caller to access individual fields within the UTOKEN.

TOKENXTR request. The issuing of the RACROUTE macro with REQUEST=TOKENXTR specified. A TOKENXTR request extracts a UTOKEN from the current address space, task or a caller-specified ACEE.

TP. See *transaction program*.

tranquility. Keeping the security classification of a resource constant while it is in use; keeping the security classification of a user constant while active.

transaction program (TP). A program that processes transactions in an SNA network.

TVTOC. See *tape volume table of contents*.

U

UACC. See *universal access authority*.

UADS. See *user attribute data set*.

universal access authority (UACC). The default access authority that applies to a resource if the user or group is not specifically permitted access to the resource, unless the user is restricted. The universal access authority can be any of the access authorities.

universal group. A user group defined using the UNIVERSAL operand of the ADDGROUP command. Universal groups are expected to have a large number of members and are unlikely to be deleted. Group profiles for universal groups do not contain complete membership information, and the LISTGRP command is not recommended to list members. Using the output of the database unload utility (IRRDBU00) is the best way to list members of a universal group.

user. A person who requires the services of a computing system.

user attribute. The extraordinary privileges, restrictions, and processing environments assigned to a user. The user attributes are SPECIAL, AUDITOR, CLAUTH, OPERATIONS, GRPACC, ADSP, and REVOKE.

user attribute data set (UADS). In TSO, a partitioned data set with a member for each authorized user. Each member contains the appropriate passwords, user identifications, account numbers, LOGON procedure names, and user characteristics that define the user.

user certificate. A type of certificate managed by RACF. See *digital certificate*.

user data set. A data set defined to RACF in which either the high-level qualifier of the data set name or the qualifier supplied by an installation exit routine is a RACF user ID.

user ID. A RACF user ID. A string of 1–8 alphanumeric characters that uniquely identifies a RACF user, procedure, or batch job to the system. For TSO users, the user ID cannot exceed 7 characters and must begin with an alphabetic, #, $, or @ character. The user ID is defined by a user profile in the RACF database and is used as the name of the profile.

user ID association. A relationship between two user IDs, established through the RACLINK command, which is required for command direction and password synchronization between the user IDs. See *peer user ID association* and *managed user ID association*.

user identification. See *user ID*.

user identification and verification. The acts of identifying and verifying a RACF-defined user to the system during logon or batch job processing. RACF identifies the user by the user ID and verifies the user by the password, PassTicket, verified digital certificate, DCE credentials, or operator identification card supplied during logon processing or the password supplied on a batch JOB statement.

user identifier (UID). A number between 0 and 2 147 483 647 that identifies a user to z/OS UNIX. The UID is associated with a RACF user ID when it is specified in the OMVS segment of the user profile. It can be contained in an object of type uid_t, that is used to identify a system user. When the identity of the user is associated with a process, a UID value is referred to as a real UID, an effective UID, or an (optional) saved set UID. See *real UID*. Contrast with *effective user identifier (effective UID)*.

user name. In RACF, 1–20 alphanumeric characters that represent a RACF-defined user. Contrast with *user ID*.

user profile. A description of a RACF-defined user that includes the user ID, user name, default group name, password, profile owner, user attributes, and other information. A user profile can include information for subsystems such as TSO and DFP.

UTOKEN. The RACF user security token. A UTOKEN is an encapsulation or representation of the security characteristics of a user. RACF assigns a UTOKEN to each user in the system. See *STOKEN* and *RTOKEN*.

V

verification. See *user identification and verification*.

VERIFY request. The issuing of the RACROUTE macro with REQUEST=VERIFY specified. A VERIFY request is used to verify the authority of a user to enter work into the system. The VERIFY request replaces the RACINIT function.

VERIFYX request. The issuing of the RACROUTE macro with REQUEST=VERIFYX specified. A VERIFYX request verifies a user and builds a UTOKEN, and handles the propagation of submitter ID.

Virtual Machine (VM). (1) An operating system that appears to be at the exclusive disposal of the particular user, but whose functions are accomplished by sharing the resources of a real data processing system. (2) In VM/ESA, the operating system that represents the virtual processors, virtual storage, virtual devices, and virtual channel subsystem allocated to a single user. A virtual machine also includes any expanded storage dedicated to it.

VM. See *Virtual Machine*.

W

workspace data sets. VSAM data sets used by RACF for queuing requests sent to and received from target nodes in an RRSF environment.

Z

z/OS. A program licensed by IBM that not only includes and integrates functions previously provided by many IBM software products, including the MVS operating system, but also:

1. Is an open, secure operating system for IBM enterprise servers
2. Complies with industry standards
3. Is based on the new 64-bit z/Architecture
4. Supports technology advances in networking server capability, parallel processing, and object-oriented programming

z/OS UNIX group identifier (GID). See *group identifier (GID)*.

z/OS UNIX System Services (z/OS UNIX). The set of functions provided by the shells, utilities, kernel, file system, debugger, Language Environment, and other elements of the z/OS operating system that allows users to write and run application programs that conform to UNIX standards.

z/OS UNIX user identifier (UID). See *user identifier (UID)*.

Index

Special Characters

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Readers' Comments — We'd Like to Hear from You

z/OS
SecureWay Security Server RACF
Security Administrator's Guide

Publication No. SA22-7683-01

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

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

Name _____ Address _____

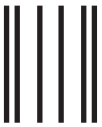
Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
SA22-7683-01

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie, NY
12601-5400

Fold and Tape | **Please do not staple** | Fold and Tape

Cut or Fold
Along Line

Program Number: 5694-A01

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SA22-7683-01