z/VM
7.4

RACF Security Server Security Administrator's Guide

SC24-6311-74

Note:

Before you use this information and the product it supports, read the information in “Notices” on page 325.

This edition applies to version 7, release 4 of IBM® z/VM® (product number 5741-A09) and to all subsequent releases and modifications until otherwise indicated in new editions.

Last updated: 2024-09-18

Contents

Figures

Tables

About This Document

This document contains information on implementing a security policy using IBM RACF® Security Server for z/VM.

Though this information is specific to z/VM, there are references to z/OS®. These references are applicable only when sharing a RACF database with a z/OS system, which is supported only on z/VM 7.2 and earlier versions.

This document helps the RACF security administrator do the following:

- Planning:
  - Decide how to use RACF to increase the security of the system.
  - Organize a security implementation team.
  - Plan whether to delegate administrative tasks (a decentralized approach) or to restrict administrative tasks to a few users (a centralized approach).
  - Plan which system resources to protect. For example:
    - System minidisks
    - SFS files and directories
    - Certain z/VM events
    - Terminals
  - Plan which user resources should be protected by security administrators and which should be protected by the users themselves. For example, RACF can protect the following user resources:
    - User minidisks
    - SFS files and directories
  - Plan which users and groups are to be known to RACF. For example, an installation can use RACF to require that all batch jobs be associated with a RACF-defined user ID.
- Daily administration:
  - Give users access to the system (assigning user IDs and passwords)
  - Give users access to system resources or functions
  - Assist users with access control problems (such as forgotten passwords or authorizations required to do their jobs).
- Coordinating with administrators of other products that use RACF for authorization.

Intended Audience

This document is intended for administrators who are responsible for controlling access to z/VM systems and their resources. In general, security administrators have the system-SPECIAL attribute, which allows them to issue any RACF command (except those that require the AUDITOR attribute). Group administrators have been granted specific authority (not usually granted to users) to perform security-related tasks related to a RACF group, or a RACF group's resources.

Note: This document should be read by RACF auditors, but is not the primary reference for them. RACF auditors should see *z/VM: RACF Security Server Auditor's Guide*.

Readers must be familiar with the RACF concepts and terminology described in *z/VM: RACF Security Server General User's Guide*. The readers of this document should also be familiar with z/VM systems.

For additional information about developing a security plan, see *z/VM: RACF Security Server Auditor's Guide*.

Most of this document describes how to protect particular kinds of resources, such as minidisks, terminals, and SFS files and directories. In general, you will first need to define users to RACF and set some RACF options. Then, depending on your security plan, you will select which classes of resources to protect, and create resource profiles for them.

Where to Find More Information

For information about related publications, refer to the “Bibliography” on page 329.

Links to Other Documents and Websites

The PDF version of this document contains links to other documents and websites. A link from this document to another document works only when both documents are in the same directory or database, and a link to a website works only if you have access to the Internet. A document link is to a specific edition. If a new edition of a linked document has been published since the publication of this document, the linked document might not be the latest edition.

How to provide feedback to IBM

We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information. See How to send feedback to IBM for additional information.

Summary of Changes for z/VM: RACF Security Server Security Administrator's Guide

This information includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations for the current edition are indicated by a vertical line (**|**) to the left of the change.

SC24-6311-74, z/VM 7.4 (September 2024)

This edition supports the general availability of z/VM 7.4. Note that the publication number suffix (-74) indicates the z/VM release to which this edition applies.

SC24-6311-73, z/VM 7.3 (December 2023)

This edition includes terminology, maintenance, and editorial changes.

SC24-6311-73, z/VM 7.3 (September 2022)

This edition supports the general availability of z/VM 7.3. Note that the publication number suffix (-73) indicates the z/VM release to which this edition applies.

SC24-6311-02, z/VM 7.2 (May 2022)

This edition includes maintenance changes.

Miscellaneous updates for May 2022

The option of issuing a warning message is updated. See “Allowing a Warning Period” on page 33.

SC24-6311-02, z/VM 7.2 (September 2020)

This edition includes changes to support the general availability of z/VM 7.2.

Chapter 1. Introduction

The security policies that protect your organization's computer resources are more important than ever. RACF provides System Administrators with the tools to implement their organizational security policies. These policies relate to authentication, authorization, and audit.

- Authentication includes:
  - User identity provisioning
  - Password management
  - Enforcement of accountability
- Authorization refers to access control lists (ACLs)
- Audit includes:
  - Recording the use of system resources
  - Recording the activities of specific users
  - Intrusion detection

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 material 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 make sure your installation's data is properly protected. RACF can help you protect this data.

How RACF Meets Security Needs

RACF can satisfy 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 managed by the z/VM Control Program (CP) and applications running in virtual machines.For a software access control mechanism to work effectively, it must be able to first *identify* the person who is trying to gain access to the system, and then *verify* that the user is really that person.

RACF has several methods available to perform its user identification and verification:

- a password
- a password phrase
- Multi-Factor Authentication (MFA)

Note: The terms *password* and *credential* are applied elsewhere in this publication to refer generically to all of these methods, unless stated otherwise.

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.

For more information, see "Overview of Authentication" on page 53.

The secured signon function provides an alternative to the authentication methods listed above, 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 Chapter 19, "Using the Secured Signon Function," on page 283.

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 2. 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, such as minidisks, terminals, and virtual switches, 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.

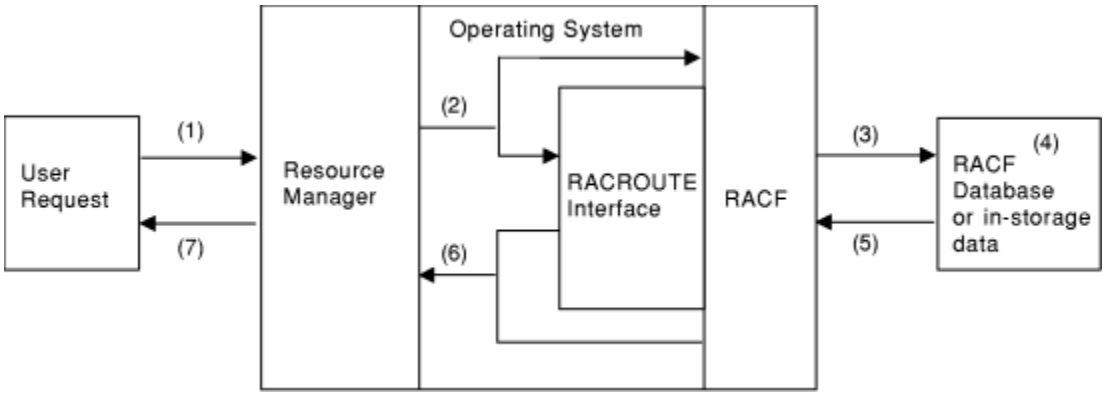


Figure 1. Authorization Checking

As shown in Figure 1 on page 2, RACF performs authorization checking as follows:

1. A user requests access to a resource through a resource manager such as SFS or the z/VM control program (CP).
2. The resource manager issues a RACF request to find out if the user can have access to the resource. In most cases, this request is a RACROUTE macro; otherwise, it is an independent RACF macro.
3. RACF refers to the RACF database or to profiles copied into storage from the RACF database.
4. RACF checks the appropriate resource profile.
5. Based on the information in the profile...
6. ...RACF passes the status of the request—the user can or cannot have access—to the resource manager.
7. The resource manager grants or denies the user's request.

Auditing and Reporting

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

- **Logging:** RACF writes audit records to a CMS file called the SMF (System Management Facility) log. All detected, unauthorized attempt to enter the system are logged. Optionally, RACF writes audit records for authorized attempts and/or detected, unauthorized attempts to:
  - Access RACF-protected resources
  - Issue RACF commands
  - Modify profiles on the RACF database.

  The audit records can be viewed using the RACF Report Writer. If you prefer, you can use the RACF SMF data unload utility (IRRADU00) to create a file that can be uploaded to a database such as DB2®. This will enable you to use database query or reporting packages to analyze the audit logs. The unload utility can also produce a file in XML format, suitable for upload to any XML-enabled spreadsheet.

  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/VM: RACF Security Server Auditor's Guide*. For information on how to specify logging and auditing functions, see the *z/VM: RACF Security Server 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 lets you send a message to a RACF-defined VM 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 a reader file that contains the notification information.

  If you are auditing access attempts, and if 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. If you are deferring access authorization to z/VM through the use of the SYSSEC macro, and are auditing access attempts, RACF writes SMF records for access attempts that would have failed if you were not deferring.
- **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 z/VM user gains access to the system through a terminal, the concept of *individual user identity* is clear. 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.) The LOGON BY function allows you to LOGON to service machines to maintain accountability. See “Defining Shared User IDs” on page 79.

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, on z/VM, a user ID can be associated with a disconnected service machine or Linux® image. In addition, in many systems today a “user” is equated with a function, rather than an individual. From the security standpoint, having users log on to application user IDs by use of a shared password is undesirable because individual accountability is lost. Instead, you should define the application user ID as a shared user ID, and require humans to use the LOGON BY function, providing their own password, when accessing the user ID. 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 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. For more information, see Chapter 5, “Defining Groups,” on page 89.

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

On z/VM systems, you can use RACF to control access to:

- The system
- Terminals
- Minidisks
- OpenExtensions resources
- SFS files and directories
- Virtual unit record devices
- Guest LANs
- Virtual switches
- VLANs
- RSCS nodes
- A subset of CP commands, DIAGNOSE codes, and system functions

- Restricted segments
- Tape volumes (when a tape management system is installed)
- Printers
- Restricted segments
- Alternate user IDs (batch processing)
- Installation-defined resources

For more information, see “Protecting General Resources” on page 15.

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:

- User's *identity*
- User's *attributes*
- User's *group authorities*
- *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 1- to 8-character user ID. With this user ID, 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 restrictions 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: On z/VM systems, you can assign the SPECIAL, AUDITOR, ROAUDIT, OPERATIONS, CLAUTH, and REVOKE attributes at the system level. When you assign attributes at the system level, the privileges and restrictions apply across the entire system. See “Assigning Optional User Attributes” on page 11 and “User Attributes” on page 57 for detailed information about these attributes.
- Group-level attributes: When you assign an attribute at the group level, RACF limits the privileges or restrictions 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). See “Assigning Optional User Attributes” on page 11 and “User Attributes” on page 57 for more information about the group-SPECIAL, group-AUDITOR, and group-OPERATIONS attributes.

Group Authorities: Each user 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, CONNECT, and JOIN.

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

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, CONNECT, and JOIN group authorities are described in detail in Chapter 5, “Defining Groups,” on page 89.

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 which 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 17, “Security Classification and Zoning,” on page 249.

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 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 ID in the resource profile's access list. Associated with each user ID or group ID 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, READ, UPDATE, CONTROL, and ALTER (see Table 38 on page 220).

For general resource profiles, an entry in the access list may also contain the name of a RACF-defined terminal. For more information, see “Conditional Access Lists for General Resource Profiles” on page 99.

Each resource also has a universal access authority (UACC) associated with it. The UACC can be NONE, READ, UPDATE, CONTROL, or ALTER. The UACC is the access authority allowed to any user or group who is not represented in the access list. UACC applies to all users, whether they are RACF-defined or not.

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
- General resource profiles

User and group profiles contain descriptions of the authorized users of a RACF-protected system; 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 data processing installation differ, RACF is designed to be 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 employ 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 RACFRW (report writer) command, and the BLKUPD command, have Interactive System Productivity Facility (ISPF) entry panels and associated help panels. These panels make it easy to enter command options on z/VM.

  You can define users and minidisks to both RACF and the z/VM directory using dual registration panels. For more information on dual registration panels, see “Using Dual Registration Panels” on page 229.
- **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/or 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 IBM-supplied classes. For more information on creating installation-defined resource classes, see *z/VM: RACF Security Server System Programmer's Guide*.

- **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 18.

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

Multilevel Security

The security policy that you implement in a multilevel-secure environment has as its key feature 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.

These are the key aspects of a multilevel-secure environment:

- Mandatory access control (MAC)
- Security labels
- Discretionary access control (DAC)
- Resource reuse
- Identification and authentication
- Auditing

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 security label. The security label is composed of a security level and zero or more security categories. The security level indicates a level or hierarchical classification of the information (for example, `Restricted`, `Confidential`, or `Internal`). The security category defines the category or group 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.

Security Labels

Security labels can be 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.

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 be granted this permission by the resource's owner.

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.

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. The following characteristics are also important:

- 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.

RACF Transparency

No users of a data processing system want their data destroyed or altered by other individuals (or by 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.

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 will be 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

While 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 will be 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 for 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 will vary 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.

Using RACF Commands or Panels

After the planning for RACF implementation has taken place (see Chapter 2, “Organizing for RACF Implementation,” on page 29 for details), you can perform security and group administration tasks, largely, 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; and so on. (Sample command sequences are given throughout this book for administrative tasks. See *z/VM: RACF Security Server 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 (data set profiles for z/OS, and user, group, 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 the ISPF panels (assuming that the ISPF product is installed at your location). If you use the panels, you need not memorize command or operand names; you need only fill in the appropriate information on the proper panels.

On z/VM systems, you can use dual registration to define users to both RACF and the z/VM directory at the same time. See “Using Dual Registration Panels” on page 229 for more information.

Choosing between RACF Commands and ISPF Panels

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

The **RACF 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:
  - To see online help for the PERMIT command when you are using the RAC command, enter:

    ```
    RAC HELP PERMIT
    ```

  - In a RACF command session, enter:

    ```
    HELP PERMIT
    ```

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

    `HELP PERMIT SYNTAX` or `RAC HELP PERMIT SYNTAX`

- If you use the RAC option, the RACF command output is displayed on the screen and also written to the RACF DATA file.

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

- ISPF creates a summary record in the ISPF log of the work you do. Unless you spool your console, the RACF commands do not create such a record. See *z/VM CMS User's Guide* for more information.
- 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 will determine whether output (for example, profile listings, search results, RACF options, and z/VM event settings) is displayed in a scrollable form.

  If your installation uses XEDIT for display in ISPF, you can save the listings in a file. You can also save the output from a SEARCH in a REXX EXEC.
- The ISPF panels for working with z/VM events provide selection lists. Using the selection lists, you can avoid typing errors when specifying RACF event names.
- The ISPF panels for working with password rules allow you to enter all the password rules on one panel.

IBM Tivoli® zSecure Manager for RACF z/VM also provides a panel interface to 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. In small installations, 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.

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 most frequent relationship is members of a department. Many installations pattern their group-user structure after their organization charts.

In the RACF group-user structure is an additional group called SYS1. When you install RACF, it will define this group for you. It 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, AUDITOR and group-AUDITOR, and ROAUDIT, and the differences among 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 RACF will use. 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 Chapter 4, “Defining Users,” on page 53. For complete descriptions of RACF commands, see *z/VM: RACF Security Server 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. As users are connected to and removed from the group over time, no changes are required to resource access lists. 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 (must be 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, restrictions, 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) when a subgroup is owned by a user, rather than a superior group. Figure 2 on page 12 shows an example of the scope of control of an attribute assigned at the group level.

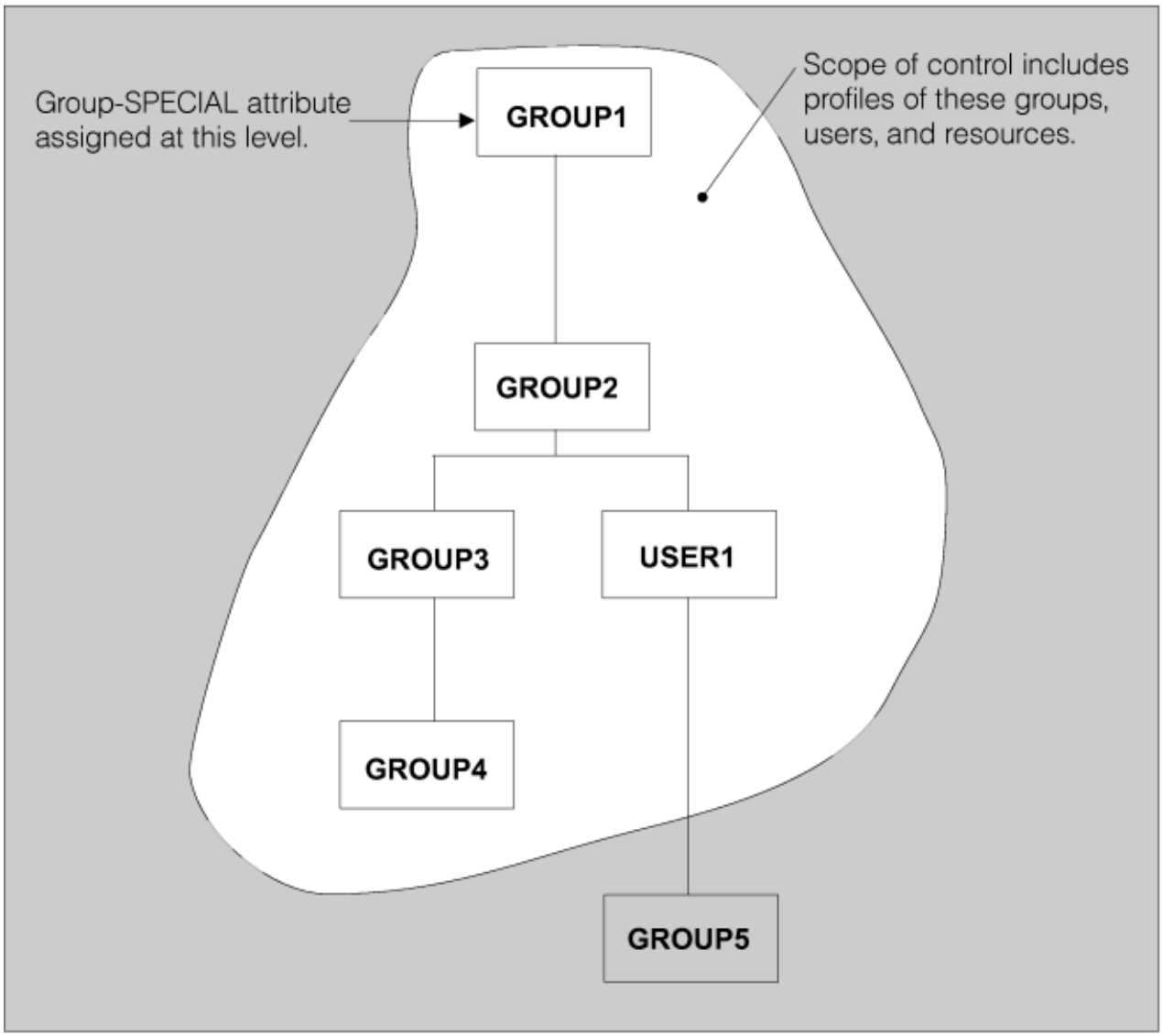


Figure 2. Scope of Control of an Attribute Assigned at the Group Level

Figure 2 on page 12 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 on page 12 lists and describes attributes that can be assigned at the user and group level. For a more complete description, see Chapter 5, "Defining Groups," on page 89.

Table 1. User Attributes

User Attribute	Description
SPECIAL	The SPECIAL attribute gives the user full control over all the RACF profiles in the RACF database when you assign it at the system level. 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 resources that are within the scope of the group, and cannot issue RACF commands that would have a global effect on RACF processing.

Table 1. User Attributes (continued)	
User Attribute	**Description**
AUDITOR	When you assign the AUDITOR attribute at the system level, it gives the user full responsibility for auditing the security controls and the use of system resources across the entire system. With it, the user can specify logging options on the RACF commands, can list the auditing options of any profiles using the RACF commands, and can control additional logging to SMF for detecting changes and attempts to change the RACF database or for detecting accesses and attempted accesses of RACF-protected resources. When you assign the AUDITOR attribute at the group level (that is, when you assign the *group-AUDITOR* attribute), authority is restricted to resources that are within the scope of the group.
ROAUDIT	When you assign it at the system level, the ROAUDIT attribute gives the user responsibility for auditing the security controls and use of system resources across the entire system without the authority to alter the system logging options. With the system-wide ROAUDIT attribute, the user can list the auditing options of any profiles using the RACF commands.
OPERATIONS	When you assign this attribute at the system level, it allows the user to perform any maintenance operations, such as copying, reorganizing, cataloging, and scratching, on RACF-protected resources. At the *group-OPERATIONS* level, authorization to perform these operations is restricted to the 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 defined in the class descriptor table. For example, IBM supplies the TERMINAL class (for terminals) and the TAPEVOL class (for tape volumes). For a list of valid class names, see “Protecting General Resources” on page 15. This authority is restricted if the SETROPTS GENERICOWNER option is in effect. See “Restricting the Creation of General Resource Profiles (GENERICOWNER Option)” on page 103.
REVOKE	This attribute excludes 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 connected to that group.

Note: You and your delegates should assign the SPECIAL, AUDITOR, ROAUDIT, 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, CONNECT, and JOIN. Each higher-level authority includes the lower levels of authority. Basically, the USE authority permits a user to access resources to which the group is authorized; the CONNECT authority enables the user to add previously RACF-defined users to the group; and the JOIN authority enables the user to define new users and new groups. See “Group Authorities” on page 92 for specific details.

Profiles Associated with Users and Groups

When you use the various 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
- 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
- The user's security level
- The user's default security label

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
- A list of connected users (members)
- The group authorities of each member

Protecting Resources

Enhancements to RACF, the z/VM operating systems and to applications have broadened the meaning of the term *resource* to include the following:

- Places in the system where data resides (such as minidisks on z/VM)
- Places in the system where data passes during data 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. On z/VM, a profile called SMITH.191 in class VMMDISK would protect SMITH's A-disk.
- *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. On z/VM, a profile called SMITH.* in class VMMDISK would protect all of SMITH's minidisks that did not have a more specific profile defined. SETROPTS GENLIST should be used when generic profiles are being used on z/VM. In most general resource classes, you can also provide a backup generic profile that protects all resources not otherwise protected.

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

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 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 which general resource class the profile is in.

IBM-Supplied Resource Classes that Apply to z/VM Systems

For a complete listing of all IBM-supplied resource classes in the CDT, see *z/VM: RACF Security Server Macros and Interfaces*.

Class	Purpose
DIRACC	Controls auditing (via SETROPTS LOGOPTIONS) for access checks for read/write access to HFS directories. Profiles are not allowed in this class.
DIRECTRY	Protection of shared file system (SFS) directories.
DIRSRCH	Controls auditing (via SETROPTS LOGOPTIONS) of HFS directory searches. Profiles are not allowed in this class.
FACILITY	Miscellaneous uses. Profiles are defined in this class so resource managers (typically program products or components) 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.
FSOBJ	Controls auditing (via SETROPTS LOGOPTIONS) for all access checks for HFS objects except directory searches. Controls auditing (via SETROPTS AUDIT) of creation and deletion of HFS objects. Profiles are not allowed in this class.
FSSEC	Controls auditing (via SETROPTS LOGOPTIONS) for changes to the security data (FSP) for HFS objects. Profiles are not allowed in this class.
GLOBAL	Global access checking. [1]
GMBR	Member class for GLOBAL class (not for use on RACF commands).
GTERMINL	Terminals with IDs that do not fit into generic profile naming conventions. [1]
PROCESS	Controls auditing (via SETROPTS LOGOPTIONS) of changes to UIDs and GIDs of OpenExtensions VM processes. Controls auditing (via SETROPTS AUDIT) of dubbing and undubbing of OpenExtensions VM processes. Profiles are not allowed in this class.
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.
RACFEVNT	RACFEVENT class contains profiles which control whether RACF change log notification is performed for USER profiles, and whether password or password phrase enveloping is to be performed.
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.

Class	Purpose
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.
SURROGAT	If surrogate submission is allowed, and if allowed, which user IDs can act as surrogates.
TAPEVOL	Tape volumes.
TERMINAL	Terminals (TSO or z/VM). See also GTERMINL class.
VMBATCH	Alternate user IDs.
VMCMD	CP commands, DIAGNOSE instructions, and system events.
VMDEV	Control who connects to real devices.
VMLAN	Use RACF to control Guest LANs
VMMAC	Used in conjunction with the SECLABEL class to provide security label authorization for some z/VM events. Profiles are not allowed in this class.
VMMDISK	z/VM minidisks.
VMNODE	RSCS nodes.
VMRDR	z/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 z/VM events) on z/VM systems.
VMPOSIX	Contains profiles used by OpenExtensions z/VM.
WRITER	z/VM print devices.

Note:

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.

IBM-Supplied Resource Classes that Apply to z/OS Systems

In order to support shared database configurations, RACF on z/VM knows about the resource classes used exclusively by z/OS. These classes appear in z/VM externals such as SETROPTS and DSMON output as well as in *z/VM: RACF Security Server Macros and Interfaces* and other RACF Security Server for z/VM publications. Refer z/OS RACF documentation for more information.

Installation-Defined Classes

Your installation can add new class descriptors, or modify or delete existing class descriptors that you have added in the installation-defined part (module ICHRRCDE) of the RACF class descriptor table (CDT).

When you define a new resource class, you may 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 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 representing the members of the group.

RACF refers to the class descriptor table whenever a class-related decision (such as, "What is the maximum length of profile names?") must be made. With the services provided by the class descriptor table, and with appropriate use of RACF authorization checking services, you can extend RACF protection to any part of your system.

See the *z/VM: RACF Security Server System Programmer's Guide* for more information on creating installation-defined classes.

Authority to Create Resource Profiles

Users can create FILE or DIRECTRY profiles if the second qualifier of the profile matches their user ID, or if they have the system-SPECIAL or group-SPECIAL attribute.

Users can create general resource profiles if they have the CLAUTH attribute for the class, or if they have the system-SPECIAL attribute.

When a resource profile is created, the creator is placed on the access list with ALTER authority. This allows the creator to access the resource(s) protected by that profile and, for discrete profiles, to manage the profile. If this is contrary to your security policy, you must establish a process whereby the access list entry is removed after the profile is created.

Note:

1. For complete descriptions of the required authorizations to any RACF command, see the description of the command in *z/VM: RACF Security Server Command Language Reference*.
2. If the SETROPTS GENERICOWNER option is in effect, further restrictions apply. See "Restricting the Creation of General Resource Profiles (GENERICOWNER Option)" on page 103.

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 FILE or DIRECTRY profiles, have a user ID that matches the second 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 FILE or DIRECTRY profiles, have a user ID that matches the second 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/VM: RACF Security Server Command Language Reference*.

Owners of Resource Profiles

In general, when you create a RACF profile, you are made 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 39 on page 221.

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 reads an SFS file with a second qualifier that matches the user's user ID, or (unless the SECLABEL class is active) when a user attempts to LINK to his or her own minidisk (identified by the MDISK statement in the user's z/VM directory entry).

Setting up the Global Access Checking (GAC) Table

You can use global access checking to improve performance of RACF authorization checking for selected resources. For example, an entry in the global access checking table can allow all users on the system to have READ access to the MAINT 190 minidisk.

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, which can result in substantial performance improvements.

Note:

1. If an entry in the global access checking table allows a requested access to a resource, no auditing is done for the request.
2. For the VMMDISK class, consider using the global disk table instead of the GAC table. The global disk table will yield a quicker result than the GAC table, but updates to the global disk table require a system IPL. See the global disk table in *z/VM: RACF Security Server Macros and Interfaces*.

For information on planning and setting up the global access checking table, see Chapter 7, “Fast Authorization Using the Global Access Checking (GAC) Table,” on page 121.

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 17, “Security Classification and Zoning,” on page 249.

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. The individual SETROPTS options are described in the chapters to which they apply. For a complete list of SETROPTS options, see *z/VM: RACF Security Server Command Language Reference*.

Using RACF Installation Exits to Customize RACF

You can tailor RACF to bypass security checking or to perform additional security processing or checking by making use of various *installation exits*. (Installation exits are perhaps more in the realm of the technical support personnel, and as such are discussed in detail in *z/VM: RACF Security Server 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.)

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

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

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 RACROUTE REQUEST=LIST processing options and after RACROUTE REQUEST=LIST processing to perform housekeeping. RACF calls the RACROUTE REQUEST=LIST selection exit to resolve conflicts between new and existing profile information.

RACROUTE REQUEST=FASTAUTH Exits

The RACROUTE REQUEST=FASTAUTH macro uses the resident profiles constructed by the REQUEST=FASTAUTH macro to perform authorization checking. The REQUEST=FASTAUTH macro has a preprocessing exit that allows you to make additional security checks or to instruct RACROUTE REQUEST=FASTAUTH to accept or fail the request to access a resource. The REQUEST=FASTAUTH macro also has a postprocessing exit that allows you to make additional security checks.

Command Exits

RACF provides exits that are called when the RACF commands ADDSD, ALTDSD, DELDSD, DELGROUP, DELUSER, LISTDSD, PERMIT, REMOVE, and SEARCH are issued. These exits permit the installation to perform additional authorization checking or to modify authorization checking when these commands are issued.

Password Processing Exit

The password processing exit supplements the processing RACF performs for new passwords and password change interval values. This exit gains control from the PASSWORD and ALTUSER commands and the RACINIT macro. The PASSWORD and ALTUSER commands and the RACINIT macro call this exit before actually changing the current password, password change interval, or both.

Password Phrase Processing Exit

The password phrase processing exit supplements the processing RACF performs for new password phrases. This exit gains control from the PASSWORD, ADDUSER, and ALTUSER commands and the RACINIT macro. The PASSWORD, ADDUSER, and ALTUSER commands and the RACINIT macro call this exit before actually changing the current password phrase.

RACF Encoding Exit

If the KDFAES (key derivation function with advanced encryption standard) algorithm is not active, the DES (data encryption standard) algorithm is used by default to encode the RACF password data that is stored in the RACF database. The ICHDEX01 exit allows you to entirely replace the RACF DES encoding routine with whatever routine your installation chooses.

For information on coding RACF installation exits, see *z/VM: RACF Security Server System Programmer's Guide*.

To see a report describing the RACF installation exits on your system, use the data security monitor (DSMON).

Tools for the Security Administrator

RACF provides a number of tools to help you (and the auditor) monitor and control RACF events.

Using the RACF Database Unload Utility

With a restructured database, you can unload the database to a sequential file by use of the database unload utility.

You can use the sequential file as input to a database management system. You can then create a relational representation of the RACF database. Effective use of the relational database provides the security administrator with installation-tailored reports.

For information on running the utility, see "Using IRRDBU00 on z/VM (RACFDBU)" on page 270. For information on using the utility output, see "Using the Database Unload Utility Output" on page 275.

Using the RACF SMF Data Unload Utility

The RACF SMF data unload utility is the IBM-recommended utility for processing RACF audit records. With it, you can create a sequential file from the security-relevant audit data. The sequential file can be:

- Viewed directly
- Used as input for installation-written programs
- Manipulated with sort/merge utilities
- Uploaded to a data manager to process complex inquiries and create installation-tailored reports.

For information on how to use the SMF data unload utility, see *z/VM: RACF Security Server Auditor's 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, which explains the meaning of the event and qualifier numbers that appear in 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.

You can find details on use of the report writer in *z/VM: RACF Security Server Auditor's Guide*.

Using the Data Security Monitor

The data security monitor (ICHDSM00, usually called DSMON) is a 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. The reports that DSMON produces are:

- System report
- RACF exits report
- Selected user attribute report
- Selected user attribute summary report
- Class descriptor table report
- Global access table report
- Group tree report

These reports will help you to (1) check the initial steps you took to establish system security, and (2) make additional security checks periodically.

For more information on the DSMON reports, see "Using DSMON" on page 226 or *z/VM: RACF Security Server 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 general resource classes, you can optionally record statistics for the following:

- The number of times that a resource 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 recorded in RACF profiles with various RACF commands.

Note: Statistics are not recorded for the following:

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

Listing User IDs or Group Names Found in the RACF Database

You can list all occurrences of a user ID or group name in the RACF database, discover the relationships between various users and groups, and learn other important information about users, groups, and the resources they control by using the RACF database cross-reference utility (IRRUT100).

To invoke the RACF database cross-reference utility, you must be defined to RACF and must have the SPECIAL (or group-SPECIAL, as applicable) attribute. However, even if you do not have the SPECIAL or group-SPECIAL attribute, you can list occurrences of your own user ID.

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

You can find complete information about the RACF cross-reference utility and other RACF utilities in *z/VM: RACF Security Server System Programmer's Guide*.

As an alternative to IRRUT100, you can use the output from the database unload utility (IRRDBU00). Refer to "Using the Database Unload Utility Output" on page 275.

Listing Information from RACF Profiles

As shown in Table 2 on page 22, You can use the LIST commands to list the contents of profiles.

To capture the output of RACF commands, do one of the following:

- In a RACF command session, spool your console.
- When using the RAC command processor, the output of RACF commands is placed in a file named RACF DATA on your disk or directory accessed as A, to which you have WRITE access. For information on how to change the defaults, see *z/VM: RACF Security Server Command Language Reference*.

Table 2. Commands to List Profile Contents

Command	Function
LISTUSER	Lists the contents of user profiles. The listing shows the owner of the profile, the user name, the default group name, the groups that a user is connected to, group authorities, the date the password was last changed, the default security label, and other information. For a complete description of this listing, see *z/VM: RACF Security Server General User's Guide*.
LISTGRP	Lists the contents of group profiles. The listing shows the owner of the group profile, the superior group name, the users connected to the group, the subgroup names, and other information.
LISTDSD	Lists the contents of DATASET profiles and allows you to determine which generic profile applies to a particular data set. The listing shows the owner of the profile, the UACC, the date the profile was created, the users and groups authorized to access the data set, your highest access authority to the data set, the security label (if there is one), and other information.
RLIST	Lists the contents of profiles for general resources such as tape volumes, z/VM minidisks, DASD volumes, and terminals. The listing shows the owner of the profile, the UACC, the date the profile was created, the users and groups authorized to access the resource, your highest access authority to the resource, the security label, and other information. For a description of this listing as it applies to minidisk profiles, see *z/VM: RACF Security Server General User's Guide*.
LFILE	Lists the contents of an SFS file profile. For a description of this listing, see *z/VM: RACF Security Server General User's Guide*.
LDIRECT	Lists the contents of an SFS directory profile. For a description of this listing, see *z/VM: RACF Security Server General User's Guide*.

As an alternative to the LIST commands, you can obtain the contents of profiles from the output of the database unload utility (IRRDBU00). See “Using the Database Unload Utility Output” on page 275.

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 23.

Note: The output of this command is in line mode unless you use ISPF panels. On z/VM, you can use the RACF DATA file generated when you use the RAC command processor.

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 file. You can easily specify RACF commands (or other commands) to be saved with the profile names. On z/VM, you can use the ISPF panel dialog to to specify RACF commands (or other commands) to be saved with the profile names and to generate an EXEC.
SRFILE	Searches the RACF database for the names of profiles in the FILE class. This command is similar to the SEARCH command.
SRDIR	Searches the RACF database for the names of profiles in the DIRECTRY class. This command is similar to the SEARCH command.

When you use these commands, your 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
- Profiles that contain a specific security category
- Profiles that contain a specific security 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.

IRRDBU00 unloads your restructured database to a sequential file. The output from the database unload utility used with a relational database manager can provide you with the ability to implement additional search criteria.

Using RACF List and Search Commands Effectively

Attention:
Using SEARCH 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, may provide the same information as the SEARCH command.

Question:
How can I tell if (or how) an SFS file or directory is protected?

Answer:
Use the LFILE or LDIRECT command, omitting both the GENERIC and NOGENERIC operands:

```
LFILE fn ft directory-name
```

or

```
LDIRECT directory-name
```

This is documented in *z/VM: RACF Security Server General User's Guide*.

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

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

```
RLIST class-name resource-name
```

Note: For resources that have grouping classes (such as terminals and DASD volumes), 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 SFS files and directories that a user can access?

Answer:
Users can normally access all SFS files and directories in their own directory structure. In RACF this means that users have full authority to all profiles in the FILE and DIRECTRY classes that have a user ID qualifier that matches their own user ID. For example, user ID ANDREW has full authority to all FILE and DIRECTRY profiles which have ANDREW as the second qualifier.

Note: If SECLABEL protection is being used for the FILE and DIRECTRY classes, users may be restricted from accessing some of their own SFS files and directories, depending on the SECLABEL they are currently logged on with.

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

   ```
   SRFILE USER(userid)
   SRDIR USER(userid)
   ```

   The name of a discrete profile identifies which file or directory it protects.

   Note: If the SRFILE or SRDIR command shows a profile that contains a RACF variable (indicated by one or more & in the name) you must list the RACFVARS profile that defines the variable. For example if you see a directory profile named POOL1:ANDREW.&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 SFS files or directories a particular generic profile protects, as in issuing the LISTDSD command with the DSNS operand for data sets. As described previously, you may use the LFILE or LDIRECT command to display the RACF profile which protects a particular SFS file or directory.
3. Find the entries in the global access checking table for the FILE and DIRECTRY classes:

   ```
   RLIST GLOBAL FILE
   RLIST GLOBAL DIRECTRY
   ```

   These entries allow all users access to files or directories 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 (similar to the steps for data sets):

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

   ```
   SEARCH CLASS(class-name) USER(userid)
   ```

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

   Note:

   a. If the resource is in a class for which there can be resource group profiles (such as GTERMINL), 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 & 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 does not provide a direct way to determine which resources a particular general resource profile protects. In general, there is no list stored on the system that shows the various existing resources RACF can check. There would have to be such a list for each general resource class—and there are well over 20 classes on z/VM alone, for such resources as terminals, z/VM minidisks, and SFS files. 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 class-name
   ```

   These entries allow all users access to resources 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 group-name
```

Question:

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

Answer:

Enter the following command:

```
LISTUSER userid
```

The output of these commands is described in more detail in the *z/VM: RACF Security Server General User's Guide*.

EXECs Supplied for Use on RACF for z/VM

RACF for z/VM provides various EXECs on the product tape to facilitate installation, for use by the system administrators, auditors and general users, and for product maintenance.

To use an EXEC, type in the name of the EXEC and press the Enter key.

Auditing EXECs

EXEC
Description

RACDSMON
Executes the data security monitor (DSMON) program in the VM environment. See *z/VM: RACF Security Server Auditor's Guide* for details of how to run this EXEC.

RACFADU
Executes the RACF SMF data unload utility. See *z/VM: RACF Security Server Auditor's Guide* for more information.

RACRPORT
Executes the RACF report writer. See *z/VM: RACF Security Server Auditor's Guide* for more information.

SMFPROF
Profile EXEC for the RACFSMF virtual machine which does the SMF switching and archiving tasks. See *z/VM: RACF Security Server Auditor's Guide* for more information.

General Use EXECs

EXEC
Description

ICHDIRMV
Allows you to rename and relocate directories, subdirectories, and underlying files in the SFS environment. You can use this EXEC in place of the CMS RENAME DIRECTORY and CMS RELOCATE DIRECTORY commands when those commands are restricted. See *z/VM: RACF Security Server Security Administrator's Guide*.

ICHSFSDF
Works in conjunction with the RAC EXEC and is called when a RACF SFS command is issued. If the file pool ID or user ID has been omitted from an SFS format name, ICHSFSDF inserts the current file pool ID or the file space into the SFS format name.

You should not modify the ICHSFSDF EXEC.

ISPF
Allows users to start an ISPF panel session if ISPF is installed. The ISPF exec shipped is a sample (ISPFRACF EXECSAMP) that customers may tailor at installation time. See *z/VM: RACF Program Directory* for more information.

RAC
Allows users to enter RACF commands without entering a RACF command session. See *z/VM: RACF Security Server Command Language Reference*.

RACFLIST
Lists profiles for resources. See *z/VM: RACF Security Server General User's Guide* for more information.

RACFPERM
Grants or revokes authority to access resources. See *z/VM: RACF Security Server General User's Guide* for more information.

RACGROUP
Determines the ACIGROUP, if any, a user belongs to. See *z/VM: RACF Security Server General User's Guide* for more information.

RACOUTP
Displays the output of a RACF command that was issued with the RAC EXEC. It can also be used to tailor the output. The output is obtained from the RACF DATA file. See *z/VM: RACF Security Server System Programmer's Guide*

RACSEC
Displays the currently active security label for a user ID. For more information, see *z/VM: RACF Security Server Security Administrator's Guide*.

RCMDRFMT
Tailors the syntax of RACF commands if they are entered with the RAC EXEC. For more information, see *z/VM: RACF Security Server System Programmer's Guide*.

Installation EXECs

The following EXECs are used during installation of RACF.

EXEC
Description

GENNUC
Generates the modified CMS which gets IPLed from the 490 disk of the RACF service machine.

RACALLOC
Allocates space on an OS-formatted minidisk for use as a RACF database.

RACDSF
OS-formats a minidisk for use as a RACF database.

RACINITD
Initializes an OS-formatted minidisk or a CMS-formatted FBA disk for use as a RACF database.

RACONFIG
Defines the configuration of the primary and backup RACF databases.

RACSETUP
Contains FILEDEFs and ACCESS commands for all the RACF databases.

RPIBLDDS
Builds or adds to a RACF database by executing the RACF statements generated by RPIDIRCT.

RPIDIRCT
Scans the CP directory and produces RACF statements to build the RACF database.

See *z/VM: RACF Security Server Security Administrator's Guide* for examples and more information.

For details on running RPIDIRCT to define OpenExtensions information, see *z/VM: RACF Security Server Security Administrator's Guide*.

Security Administration EXECs

EXEC
Description

ICHSFS
Migrates SFS installations to RACF. See *z/VM: RACF Security Server Security Administrator's Guide*.

RACFDBU
Used to unload the RACF database to a sequential file. For a description of how to use RACFDBU, see *z/VM: RACF Security Server Security Administrator's Guide*.

RACFDEL
Uses the output from IRRUT100 to generate commands designed to remove occurrences of a user ID from the RACF database. For a description of how to use RACFDEL, see *z/VM: RACF Security Server Security Administrator's Guide*.

RPIDELU
Used to execute the commands generated by RACFDEL. For a description of how to use RPIDELU, see *z/VM: RACF Security Server Security Administrator's Guide*.

System Programming EXECs

EXEC
Description

RACIPLXI
Automatically logs on the AUTOLOG2 virtual machine following RACF initialization. RACF only invokes this EXEC during initialization. It can be modified to issue messages appropriate to your installation.

RACFCONV
Updates the database templates. Executes the RACF utility IRRMIN00. See *z/VM: RACF Security Server System Programmer's Guide* for more information.

RACFSVRS
Used to initialize multiple RACF service machines. See *z/VM: RACF Security Server System Programmer's Guide*

RACSTART
Starts RACF. Issued from the RACF service machine.

RACSVRXI
Exercises control following an IUCV SEVER from CP on the CP to RACF path. The RACF service machine is the only machine that invokes this EXEC, but it can be modified to issue messages appropriate to your installation.

RACUT100
Lists all occurrences of a user ID or group name in a RACF database. Executes the RACF utility IRRUT100. See *z/VM: RACF Security Server System Programmer's Guide* for more information.

RACUT200
Copies a RACF database and identifies inconsistencies in the database. Executes the RACF utility IRRUT200. See *z/VM: RACF Security Server System Programmer's Guide* .

RACUT400
Splits, merges, and copies a RACF database. Executes the RACF utility IRRUT400. See *z/VM: RACF Security Server System Programmer's Guide* for more information.

Using SMAPI with RACF

System Management Application Programming Interface (SMAPI) calls can be managed by RACF. That is, SMAPI requests can be authorized and reported on using RACF. For more information, see the following sections in *z/VM: Systems Management Application Programming*:

- In the "Setting Up and Configuring the Server Environment" chapter, see the section "Configuring an ESM for SMAPI Authorization Decisions".
- In the "Using SMAPI with an External Security Manager" appendix, see the section "Using SMAPI with RACF".

Chapter 2. Organizing for RACF Implementation

This publication 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 will not, by itself, be 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 failure of the security undertaking.

To be successful, a security implementation requires a management involved with questions of security policy, procedures, resources to be allocated to the security function, and accountability of users of the computer system. Without such management support, the security procedures will fall into disuse and will 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
- Statement of policy concerning outside users of the system
- Security attitudes expected from all users of the system

The resultant security policy will help 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
- Installation organization
- Installation standards
- Major application areas

As security administrator, you will 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

A typical list of implementation team members and their responsibilities is shown in Table 4 on page 30.

Table 4. Participants of the 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. (For example, on z/VM systems, you will need to have someone look at what security is currently defined in the z/VM directory and decide if that security is sufficient for your needs under RACF.) 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 your installation has both z/OS and z/VM and you are using RACF on both systems, you should ensure that technical personnel and other representatives of both systems are members of the implementation team. For more information, see *z/VM: RACF Security Server System Programmer's Guide*.
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/VM: RACF Security Server Auditor's Guide*. This publication outlines the procedures that a system auditor should follow and describes the RACF report writer and the data security monitor.
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: • On z/VM systems, the system administrator, an RSCS system programmer, or a PSF system programmer.

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 will need to set priorities about which data, applications, and users need to be secured. The implementation team should plan to phase in the security controls over a period of time to give the users a necessary period for adjustment.

The implementation plan should identify the major RACF events—when each must be completed, who will be responsible for each event, and interdependencies among events. In addition, the plan should take into account other significant activity planned during the same time period that could affect the implementation (new systems, hardware, applications, and so on). At an early stage it should also define a pilot group for whom protection of business data, jobs, and users, will be completed before undertaking 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, while 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. Nonconfidential data, which is recoverable with little inconvenience if destroyed
3. The vast amount of data that falls between these two extremes, which should be protected from inadvertent or deliberate modification or destruction

The data in category 1 *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, once 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 will need to 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 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 with *, **, or % in the profile name to protect them.

  For certain classes, such as terminals, you can create resource grouping profiles to protect resources using names that do not lend themselves to the use of *, **, or %.

For any general resource class, you can define a "RACF variable" that can be used in profile names in general resource classes. See “Choosing Among Generic, Resource Group, and RACFVARS Profiles” on page 102.

- On z/VM, you can protect existing minidisks by using the RDEFINE command to create profiles in the VMMDISK class. It is recommended that your installation protect existing minidisks when first installing RACF, by using the RPIDIRCT EXEC. The RPIDIRCT EXEC scans the CP directory and translates directory entries into RACF commands, including an RDEFINE command for each minidisk defined in the CP directory. This procedure is described in *z/VM: RACF Program Directory*.

  You can use RPIDIRCT to scan the directory for OpenExtensions information. See “Using RPIDIRCT to Prime the RACF Database from the CP Directory” on page 44 for more information.

  You can also use ICHSFS to migrate existing SFS file and directory protection into RACF profiles. See “Step 1: Migrate Existing SFS Authorities Into RACF Using ICHSFS” on page 182 for more information.

Note: You will need to 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 will need to create discrete profiles.

Protecting New Data

To help you protect new data, RACF provides *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. See “Protecting General Resources with Generic Profiles” on page 100 for more information.

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 profile when defining a new profile. (The copied profile is not necessarily identical to the model profile. See “Possible Changes to Copied Profiles When Modeling Occurs” on page 32.) This copying greatly reduces the effort needed to create new profiles. The following are some ways to use profile modeling.

- A user can use the FROM operand (and related operands) on the RDEFINE, ADDFILE, and ADDDIR commands to copy information from an existing profile into the new profile. RACF uses the specified profile as a model when creating the new profile. However, profile segment information is not copied to the new profile.
- A user can use the FROM operand (and related operands) on the PERMIT, PERMFILE, and PERMDIR commands to copy the access list from one existing profile into another existing profile.
- If the preceding methods are not sufficient, an installation can also use a RACROUTE 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 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, see "SETROPTS Options Related to Security Labels" on page 259.

  Exception: 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, see "SETROPTS Options Related to Security Labels" on page 259.
- For TAPEVOL profiles, TVTOC information is not copied to the new profile.

Allowing a Warning Period

In addition to deciding what to protect, the implementation team will need to 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 (specifying GLOBALAUDIT(ALL) for the resource profile) or auditing all protected resources in a class (entering the SETROPTS LOGOPTIONS command). These commands will cause SMF logging to occur for all accesses. If the profiles allow all access, the SMF records will 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 indicating on the ADDSD, RDEFINE, ADDFILE, ADDDIR, ALTDSD, RALTER, ALTFILE, or ALTDIR command that a WARNING is desired. 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 access to the resource. However, z/VM CP initiated requests (for example, LINK commands) are either deferred to CP or rejected, depending on the configuration of the SYSSEC macro.

Note: 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 command, depending on the options chosen by the resource manager issuing the macro.

On z/VM, in addition to using the warning message facility, you can use the SYSSEC macro to defer access authorization decisions to z/VM for some frequently used resource classes until you have created profiles for those resources. The SYSSEC macro is coded in RACF module HCPRWA. For more information on operating your system in this way, see the description of the SYSSEC macro in *z/VM: RACF Security Server Macros and Interfaces*.

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 simply 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.

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, while 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 (with the IBM-supplied SYS1 group as the highest group), so that each group is a subgroup of some other group. You should document the resulting group structure as part of the implementation plan. Perhaps you might want to develop a set of guidelines for your delegated security and group administrators to identify the general categories of resources, users, and the relationships between them.

Figure 3 on page 34 shows relationships that can exist between users and groups.

Note: Although this figure uses z/OS terminology to illustrate relationships between users and groups and their resources, the same concepts apply to relationships between users and groups and their resources on z/VM systems.

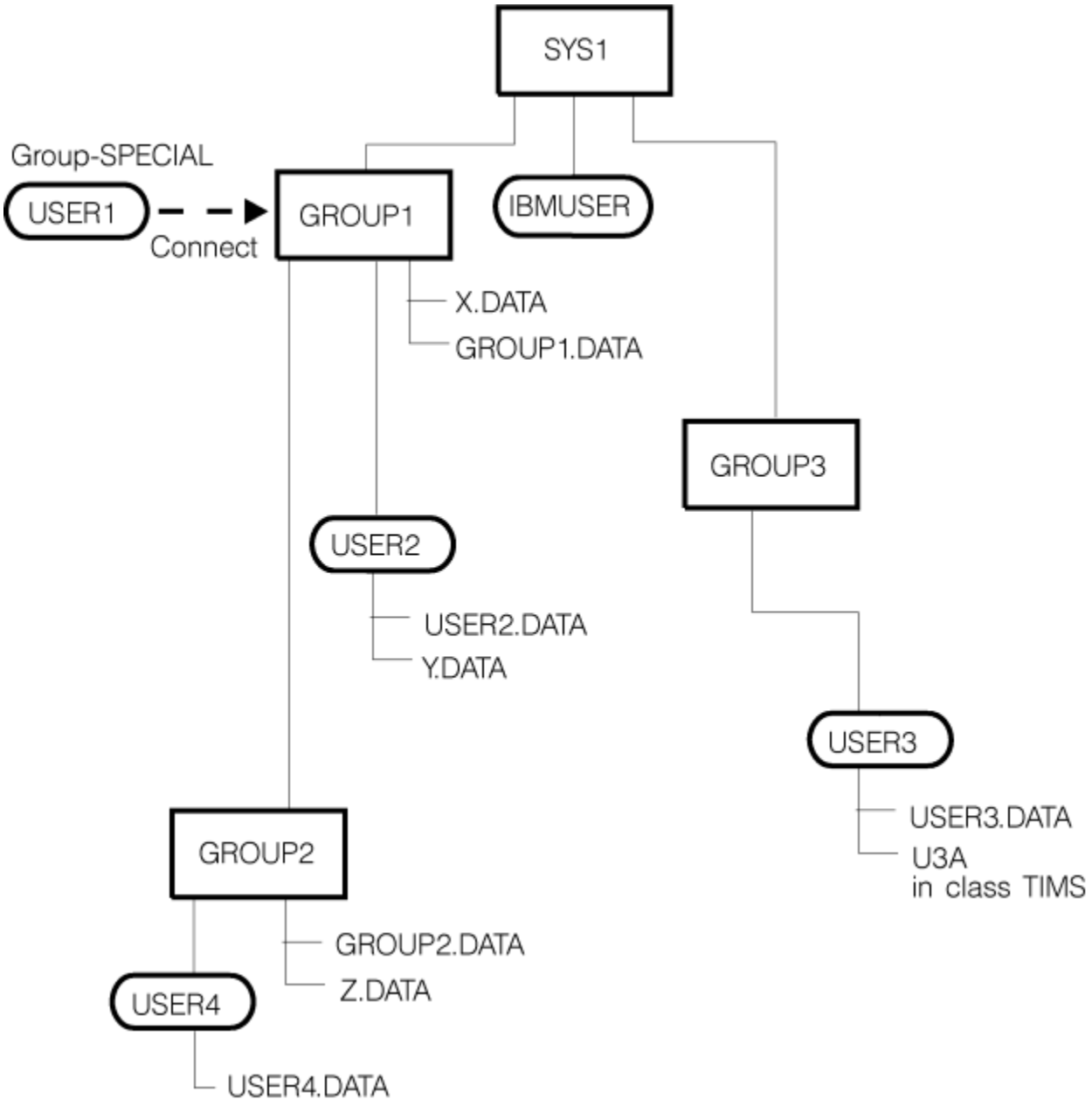


Figure 3. User and Group Relationships

In Figure 3 on page 34, the highest level group, SYS1, owns subgroups GROUP1 and GROUP3 and the user, IBMUSER. GROUP1, in turn, owns subgroup GROUP2 and the users USER1 and USER2. Note that 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 GROUP3's resources.

Note: If you run with list-of-groups checking inactive (such as, SETROPTS NOGRPLIST), then 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 "List-of-Groups Authority Checking" on page 94.

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 each user needs to know about RACF depends on the RACF functions you authorize the person to use. Some examples of information required by various types of system users are:

All System Users: All users defined to RACF must know the following:

- How to identify themselves to the system with their user ID and credential (which can be an MFA credential, a password, a passphrase, or a pass ticket).
- How to create or change their credential, depending on their authentication method.

  Users might need to be familiar with the RACF PASSWORD command if you want them to be able to change their password intervals.
- Users using MFA need to know how many times a generated credential can be used and for how long the credential is valid.
- The significance of their credential to system security.

  z/VM users and administrators should be aware that, when logging on with RACF, a user can change a current password/passphrase or establish a new one when the old one has expired. Prior to installing RACF, these options were not available to the user.
- z/VM users defined as MFA (the PWFALLBACK option) can change their password/phrase with the RACF PASSWORD command or when they LOGON with the FALLBACK option when their password/passphrase is expired.

Make sure to inform all users that they can use either the RAC command processor or the RACF command session. In addition, users should be able to:

- Use the RACF LISTUSER command to list their own profile information
- Protect their own z/VM minidisks with RACF profiles
- Protect their own SFS files and directories with RACF profiles
- Protect their own OpenExtensions resources as documented in *OpenExtensions for z/VM: User's Guide.*

If security labels are used on your system, users should know how to log on with a security label other than their default security label. For more information, see "How Users Specify Current Security Labels" on page 256.

Note: Users can enter the following command to find out what security labels they can use:

```
SEARCH CLASS(SECLABEL)
```

For complete information on tasks general users might perform using RACF, such as permitting others to use their minidisks, see *z/VM: RACF Security Server General User's Guide*.

Users Who RACF-Protect General Resources: Depending on your security plan, users might work with profiles in the VMMDISK 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 IDs they can use when giving access to the profiles
- The meaning of the access authorities (NONE, READ, and so forth) in the general resource class
- What your installation's security policy is towards specific security enhancements like security levels, categories, and security labels.

For more information, see Chapter 6, “Defining Resources,” on page 97 and the sections of this book that describe how to use the class.

Technical Support Personnel: Users who install RACF need to be familiar with migration considerations and the steps required to install or re-install RACF. See *z/VM: RACF Program Directory*.

Users who maintain the RACF database (for example, technical support personnel) must be familiar with the RACF utilities. *z/VM: RACF Security Server System Programmer's Guide* describes these utilities.

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

RACF Auditors: Users with the AUDITOR or ROAUDIT attribute should refer to *z/VM: RACF Security Server Auditor's Guide* for information on using RACF for auditing.

Note that if ISPF is 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 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, might indicate the need for more user education. The report can also help you to identify any specific users who are persistently trying to alter profiles without the proper authority.

z/VM: RACF Security Server Command Language Reference contains detailed information on the RACF commands.

Programmers Writing Unauthorized Applications: Unauthorized applications need READ access to ICHCONN in the FACILITY class. That is, any RACROUTE request which could execute without APF authority on a z/OS system requires READ access to ICHCONN. See *z/VM: Security Server RACROUTE Macro Reference* for more information on access levels required for different request types.

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 the application invokes the RACROUTE REQUEST=AUTH macro. For more information on creating installation-defined classes, see *z/VM: RACF Security Server System Programmer's Guide*.

Programmers Writing Authorized Applications: Programmers writing authorized applications can use the RACROUTE macro to request more sensitive security-related services, including:

- User identification and verification (RACROUTE REQUEST=VERIFY)
- Replacing or retrieving fields in RACF profiles (RACROUTE REQUEST=EXTRACT)

That is, any RACROUTE request which could execute only with APF authority on a z/OS system requires UPDATE access to ICHCONN. For more information on using the RACROUTE macro, see *z/VM: Security Server RACROUTE Macro Reference*.

Checklist for Implementation Team Activities

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. In some cases, you will have to actively solicit management support.

You should examine organizational structures to establish the most efficient profile ownership structures, educate users with the level of material 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 reflect the work of the team. Table 5 on page 37 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? Which resources within these classes are to be protected? Can protection be phased in? Which resources need to be protected when?
Naming conventions	What installation general resource naming conventions exist? Are changes necessary? Will implementing RACF provide an opportunity to enforce naming conventions? And if so, installation-wide, or just a subset? Immediately or eventually?
Organization	Can the definition of RACF groups (and their associated users) be molded to map into 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 are to be delegated, wholly or in part? Which users should have what RACF attributes?
User and group names	Establish names for groups and user IDs. Determine which are to be defined to RACF. Select the user verification technique.
Transparency	Try to make RACF transparent to your users wherever possible. Consider which resources can be protected by generic profiles and which resources will require discrete profiles. Determine which users and groups should be placed in the access lists, and with what access authorities. Determine what deviations from strict user accountability are to be allowed, and for how long?
RACF tailoring	Determine which RACF exits are to be used, if any, and under what conditions.
VM authorizations	Review z/VM authorizations, such as: • Minidisk links • Access to RSCS network nodes • OpenExtensions authorizations • SFS authorizations.
Recovery	Establish recovery procedures.
Violation procedures	Establish security procedures for logging, reporting, auditing.
Test plan	Develop a RACF test plan.

Table 5. Checklist for Implementation Team Activities (continued)

Item	Comments
Education	Plan to prepare user documentation and/or other educational material—perhaps a newsletter for most users, perhaps more detailed education for group administrators.
Installing RACF	Select RACF options and install RACF.
Monitoring	After beginning to define groups, users, generic profiles, and data for a pilot group, monitor progress against your implementation plan. Establish procedures to ensure that future applications receive the appropriate security considerations.

Chapter 3. Preparing to Use RACF

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)

And 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)

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 system-SPECIAL attribute, which allows IBMUSER to issue most of the RACF commands (except those reserved for users with the AUDITOR attribute) and the system-OPERATIONS attribute, which allows IBMUSER to access many RACF-protected resources.

When entering 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, enter a RACF command session:

```
RACF
```

Next, list the system-wide RACF options that are in effect:

```
SETROPTS LIST
```

Not all options are displayed at this point, because IBMUSER does not have the AUDITOR or ROAUDIT attribute. If you want to see the status of those options, grant IBMUSER the ROAUDIT attribute, log off, and log on again. To see all of the options, issue SETROPTS LIST again.

Read through the list of parameters to familiarize yourself with what options are in effect and what they mean.

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 will not be able to access your terminals and, consequently, you will be locked out of your system.

You should also check which z/VM events are controlled or audited by issuing the following command:

```
SETEVENT LIST
```

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 will also act as the system-wide auditor, you should also give this user ID the AUDITOR attribute. Do one of the following:

```
ADDUSER RACFADM PASSWORD(password)SPECIAL OPERATIONS
```

or:

```
ADDUSER RACFADM PASSWORD(password) SPECIAL OPERATIONS AUDITOR
```

Note: Make sure the new administrator has access to the minidisk that contains the RPIDIRCT SYSUT1 file. For example:

```
RDEFINE  VMMDISK  RACFVM.301
PERMIT  RACFVM.301  CLASS(VMMDISK)  ID(RACFADM)  ACCESS(UPDATE)
```

Then, exit the RACF command session:

```
END
```

Logging On as RACFADM, Checking Groups and Users, and Revoking IBMUSER

Log on as RACFADM.

Enter a RACF command session:

```
RACF
```

At this point, you will receive a message stating that your password expired. Immediately change the password to a new password.

First, list all users to ensure that only RACFADM and IBMUSER are defined to RACF, and that they have the proper attributes.

```
LISTUSER *
```

Then, list all the groups defined to RACF:

```
LISTGRP *
```

Connect RACFADM to them and make RACFADM the owner, for example:

```
CONNECT RACFADM GROUP(SYS1) AUTH(JOIN)
ALTGROUP SYS1 OWNER(RACFADM)
```

To verify that the changes took place, issue the following command:

```
LISTGRP SYS1
```

Then, revoke IBMUSER so that another user cannot make use of the IBMUSER user ID:

```
ALTUSER IBMUSER REVOKE
```

Note: You cannot delete the IBMUSER user profile.

Define another user to RACF (user ID RACFAD2), who will 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
```

Setting RACF Options

This section provides a roadmap of the system-wide options that you should consider setting when RACF is first installed.

Table 6. Recommended SETROPTS options at RACF installation

Perform this task	**Using this SETROPTS operand**
“Allowing Mixed-Case Passwords” on page 69	PASSWORD(MIXEDCASE) operand
“Allowing Special Characters in Passwords” on page 69	PASSWORD(SPECIALCHARS) operand
“Password Syntax Rules” on page 68	PASSWORD(RULE*n*) operand
“Setting the Maximum and Minimum Change Interval” on page 71	PASSWORD(INTERVAL) operand
“Extended Password Processing” on page 72	PASSWORD(WARNING/HISTORY/REVOKE) operand
“Encryption of RACF User Passwords” on page 73	PASSWORD(ALGORITHM) operand
“Revoking Unused User IDs” on page 77	INACTIVE operand
“List-of-Groups Authority Checking” on page 94	GRPLIST operand
“Setting the RVARY Passwords” on page 41	RVARYPW(SWITCH/STATUS) operand
“Restricting the Creation of General Resource Profiles (GENERICOWNER Option)” on page 103	GENERICOWNER operand
“Activating and Deactivating General Resource Classes” on page 99	CLASSACT operand
“Generic Profile Checking of General Resources” on page 104	GENERIC operand
“Using SETROPTS STATISTICS for Statistics Collection” on page 115	STATISTICS operand
“Collecting LOGON Statistics” on page 77	INITSTATS operand

Setting the RVARY Passwords

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 or change RACF status (activate/deactivate RACF). 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 a RACF database is first initialized using IRRMIN00, the switch password and the status password are both set to YES.

Defining the Groups Needed for the First Users

At this point you should consider creating the groups that you will need. You proceed to add four groups: three (GROUP1, GROUP2, and GROUP3) are departmental groups, with GROUP2 and GROUP3 owned by GROUP1 so that certain authorities can be propagated. The fourth group (DATAMGT) has minidisk maintenance responsibility.

```
ADDGROUP (GROUP1 DATAMGT)
ADDGROUP (GROUP2 GROUP3) OWNER(GROUP1) SUPGROUP(GROUP1)
```

Defining a System-Wide Auditor

Define a user who will have system-wide auditing responsibilities and privileges.

```
ADDUSER AUDCCC AUDITOR
```

Defining a System-Wide Read-Only Auditor

Define a user (AUDMON, for example) who has system-wide responsibility and privilege to monitor and view system audit information using the existing system auditing controls. Such a user would be useful in assuring system integrity without being able to alter the system auditing controls. For example:

```
ADDUSER AUDMON ROAUDIT
```

Defining Users and Groups

You now add a user (D03DIK) to GROUP3.

```
ADDUSER D03DIK OWNER(GROUP3) DFLTGRP(GROUP3)
```

The password is given by the security administrator, and this initial password is the name of the default group.

Note: For more information on defining users, see “Summary of Steps for Defining Users on z/VM” on page 84.

Defining Group Administrators, Group Auditor, and Data Manager

Define a group administrator with the group-SPECIAL attribute for each group. Only the administrator for GROUP1 has authority to define new users in that group. Each of the other administrators has authority over resources owned by his or her group, as well as 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) SPECIAL
ADDUSER D03ABL DFLTGRP(GROUP3) DATA('GROUP3 ADM')
CONNECT D03ABL GROUP(GROUP3) SPECIAL
```

Define group-auditor for groups GROUP1, GROUP2, and GROUP3. 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 z/VM minidisks to RACF in order to perform dump, restore, and data cleanup operations.

```
ADDUSER DMGJFS DFLTGRP(DATAMGT) AUTH(JOIN)
        CLAUTH(USER VMMDISK) DATA('DATA MGT ADM')
```

Because of their duties, data managers are connected to SYS1, allowing them to access resources with SYS1 in their access list. The data manager has the group-SPECIAL attribute in group SYS1.

```
CONNECT DMGJFS GROUP(SYS1) UACC(READ) SPECIAL
```

Exit the RACF command session:

```
END
```

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 (USE)
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 (USE)
GROUP3	SYS1	GROUP1	D03ABL (USE)
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 (USE)	CLAUTH(USER VMMDISK), SPECIAL, OPERATIONS	SYS1 (USE)
D01RHG	GROUP1 (JOIN)	CLAUTH(USER), group-SPECIAL	
D02JMP	GROUP2 (USE)	group-SPECIAL	
D03ABL	GROUP3 (USE)	group-SPECIAL	
D01GPB	GROUP1 (USE)	group-AUDITOR	
D03DIK	GROUP3 (USE)		
AUDCCC	SYS1 (USE)	AUDITOR	
AUDMON	SYS1 (USE)	ROAUDIT	

Using RPIDIRCT to Prime the RACF Database from the CP Directory

The RPIDIRCT EXEC helps you to migrate existing CP directory data to a RACF database. It scans the CP directory and translates directory statements into RACF commands. It places the RACF commands in an output file called RPIDIRCT SYSUT1. You can use this file to initialize new RACF databases if you are not planning on sharing an existing RACF database, or to modify an existing database if you are planning on sharing an existing RACF database.

Before you run RPIDIRCT, you might need to make some changes to the CP directory. After you run RPIDIRCT, you might need to make some changes to the RPIDIRCT SYSUT1 file. The following sections have information on CP directory requirements, the RPIDIRCT SYSUT1 file and running the RPIDIRCT EXEC.

Note: The supplied RPIDIRCT EXEC is located on the PMAINT 551 disk.

General CP Directory Requirements

Before running RPIDIRCT, review the CP directory for any of the following:

- Duplicate user IDs
- Reserved passwords
- Unacceptable characters in user IDs
- ACIGROUP statements
- Group names on POSIXGROUP statements that are duplicates except for case
- OpenExtensions-related entries added by the DIRPOSIX EXEC
- Very large directory entries
- Networking Access and Considerations

Duplicate user IDs

If you plan to have multiple z/VM systems share the RACF database, review their CP directories for cases where a user on one system has the same user ID as a different user on another system. If you find cases where this occurs, you must change one of the user IDs.

Reserved passwords

Some user passwords in the CP directory are regarded as reserved passwords and will be handled as follows.

AUTOONLY / LBYONLY / NOPASS

A user with a password of AUTOONLY, LBYONLY, or NOPASS will be defined as a protected user ID by having the NOPASSWORD, NOPHRASE, and NOMFA attributes set. If you find cases where these passwords are specified, no action is required but it is recommended that you review the information in “PROTECTED Attribute” on page 60 and “Passwords and Password Phrases” on page 67, with special attention to Note “1” on page 68.

NOLOG

A user with a password of NOLOG can be handled in one of two ways.

Allow the password to remain as NOLOG.

The user will be defined as a protected and revoked user by having the NOPASSWORD, NOPHRASE, NOMFA, and REVOKE attributes set. z/VM will not permit those users to log on to the system.

Change the password to UNLOG.

The user will be defined as a protected and revoked user by having the NOPASSWORD, NOPHRASE, NOMFA, and REVOKE attributes set. z/VM will pass the LOGON request to RACF, which will fail the attempt. Later, their ability to log on can be resumed using RACF controls, without the need for a corresponding CP directory change. See “PROTECTED Attribute” on page 60 and “REVOKE Attribute” on page 60 for more information.

Unacceptable Characters in User IDs

RACF does not allow a dash (-), plus (+), colon (:), or underscore (_) in a user ID. Remove these characters from all user IDs that contain them.

RPIDIRCT changes any unacceptable character it finds in a user ID to a dollar sign ($) but depending on the 3270 code page, they may not appear as such.

ACIGROUP Considerations

IBM recommends that the ACIGROUP statement not be used in the USER DIRECT file. RACF has its own concept of user groups which is much more robust and flexible than is the use of ACIGROUP. As such, it is recommended that you remove any ACIGROUP statements prior to running RPIDIRCT.

RPIDIRCT will generate different commands based on whether or not ACIGROUP is used. See Table 7 on page 46 for details.

Group Names on POSIXGROUP Statements that Are Duplicates Except for Case

Although VM supports mixed-case group names, RACF does not. For example, if your CP directory contains the statements:

```
POSIXGROUP Payroll 10
POSIXGROUP payroll 20
```

RACF creates a PAYROLL group profile, assigning it a GID value of 10. RPIDIRCT issues a warning message when it encounters the second POSIXGROUP statement.

OpenExtensions-Related Entries Added by the DIRPOSIX EXEC

When you run the DIRPOSIX utility with the SYSENTRIES option, the utility creates entries that RACF does not support. If you have run this utility, you might need to change some UID and GID values, and some group names. For more information, see “DIRPOSIX Considerations” on page 51.

Very Large Directory Entries

If your CP directory contains any very large entries, such as entries with several hundreds of MDISK statements, it is possible that RPIDIRCT will fail with the following error message:

```
FPLDSK078E Record length nnnnn is too much
```

where *nnnnn* is a number greater than 65535.

If this occurs, you need to remove or comment out all or part of the very large entries in the CP directory file to allow RPIDIRCT processing to complete successfully. For those entries that were either removed or commented out (either all or in part), you will need to manually create the RACF commands that would have been generated by what was removed or commented out. These RACF commands can be added to the RPIDIRCT SYSUT1 output file after RPIDIRCT is run.

Networking Access and Considerations

If your CP directory entry contains networking authorizations via the NICDEF or SPECIAL statements, RPIDIRCT will grant access for that virtual machine based upon the definitions provided. It will not

validate whether access to any particular VLAN is appropriate, or whether the Virtual Switch or Guest LAN specified exists. RPIDIRCT will do neither syntax checking or semantic checking of network configuration. It is highly recommended that you execute CPSYNTAX against a given user directory entry in order to validate correctness before running RPIDIRCT.

Additionally, be advised that PROMISCUOUS mode is enabled on a per-vdev level, which will grant access to all specified Virtual LANs associated with that base networking profile. PROMISCUOUS mode enables CONTROL access to that Guest LAN or Virtual Switch and should only be granted to appropriate virtual machines.

The RPIDIRCT SYSUT1 File

The commands generated by the RPIDIRCT EXEC are placed in an output file called RPIDIRCT SYSUT1. The commands can:

- Add a user to the RACF database
- Define a user's minidisks (MDISK statement in the CP directory)
- Define a user's dedicated devices (DEDICATE statement in the CP directory)
- Define a reader for each VM user, permitting the user to access the reader
- Define access to batch service machines
- Permit access for users having LINK statements in the directory
- Permit network access for users having NICDEF or SPECIAL statements in the directory
- Define OpenExtensions information

The objectives of your installation might require you to change some of the commands.

If you have a large installation, you might want to break up the RPIDIRCT SYSUT1 file into several smaller files. In that way, several jobs of a more manageable size can be run to initialize the RACF database. For example, you could place all the ADDUSER commands in one file, all the RDEFINE commands in another file, and the PERMIT commands in still another file. If you do this, remember to run the ADDUSER command file first, then the RDEFINE command file, and lastly, the PERMIT command files.

The general format of the commands generated by the EXEC is is shown in Table 7 on page 46.

Table 7. RPIDIRCT-generated RACF commands

Statement	**Command**
DIRECTORY	RDEFINE VMCMD RACF UACC(READ) RDEFINE VMCMD RAC UACC(READ)

Table 7. RPIDIRCT-generated RACF commands (continued)

Statement	Command
USER	ADDUSER *userid* DFLTGRP(SYS1) UACC(NONE) PASSWORD(*password*) RDEFINE VMBATCH *userid* OWNER(*userid*) UACC(NONE) PERMIT *userid* CLASS(VMBATCH) ACCESS(ALTER) RESET RDEFINE VMRDR *userid* UACC(NONE) OWNER(*userid*) PERMIT *userid* CLASS(VMRDR) ID(*userid*) ACCESS(ALTER) RESET or, if an ACIGROUP statement is present, ADDUSER *userid* DFLTGRP(*acigroup*) UACC(NONE) PASSWORD(*password*) RDEFINE VMBATCH *acigroup.userid* OWNER(*userid*) UACC(NONE) PERMIT *acigroup.userid* CLASS(VMBATCH) ACCESS(ALTER) RESET RDEFINE VMRDR *acigroup.userid* OWNER(*userid*) UACC(NONE) PERMIT *acigroup.userid* CLASS(VMRDR) ID(*userid*) ACCESS(ALTER) RESET When a CP directory password of AUTOONLY, LBYONLY, or NOPASS is encountered, the following command is generated immediately below the ADDUSER command: ALTUSER *userid* NOPASSWORD NOPHRASE When a CP directory password of NOLOG or UNLOG is encountered, the following command is generated immediately below the ADDUSER command: ALTUSER *userid* NOPASSWORD NOPHRASE REVOKE
MDISK	RDEFINE VMMDISK *userid.addr* OWNER(*userid*) UACC(MODE) PERMIT *userid.addr* CLASS(VMMDISK) RESET ID(*userid*) ACCESS(ALTER) or, if an ACIGROUP statement is present, REDEFINE VMMDISK *acigroup.userid.addr* OWNER(*userid*) UACC(MODE) PERMIT *acigroup.userid.addr* CLASS(VMMDISK) RESET ID(*userid*) ACCESS(ALTER)
LINK	PERMIT *owner.addr* CLASS(VMMDISK) ID(*userid*) ACCESS(*mode*) or, if an ACIGROUP statement is present, PERMIT *acigroup.owner.addr* CLASS(VMMDISK) ID(*userid*) ACCESS(READ)
The first occurrence of: ACIGROUP groupname in the CP directory	ADDGROUP *acigroup*
LOGONBY byuserid	RDEFINE SURROGAT LOGONBY.*userid* UACC(NONE) PERMIT LOGONBY.*userid* CLASS(SURROGAT) ID(*byuserid*) ACCESS(READ)
DEDICATE	RDEFINE VMDEV RDEV.*devno.sysname* OWNER(*userid*) UACC(NONE) PERMIT RDEV.*devno.sysname* CLASS(VMDEV) RESET ID(*userid*) AC(*access*) or, if an ACIGROUP statement is present, RDEFINE VMDEV *acigroup*.RDEV.*devno.sysname* OWNER(*userid*) UACC(NONE) PERMIT *acigroup*.RDEV.*devno.sysname* CLASS(VMDEV) RESET ID(*userid*) AC(access)

Table 7. RPIDIRCT-generated RACF commands (continued)	
Statement	**Command**
NICDEF	RDEFINE VMLAN.*owner.name* UACC(NONE) PERMIT *owner.name* CLASS(VMLAN) ID(*userid*) ACCESS(access) or, if the VLAN keyword is present on the NICDEF statement, PERMIT *owner.name.vlanid* CLASS(VMLAN) ID(*userid*) ACCESS(access)
SPECIAL	RDEFINE VMLAN.*owner.name* UACC(NONE) PERMIT *owner.name* CLASS(VMLAN) ID(*userid*) ACCESS(access)

Note: See Table 8 on page 50 for POSIX information.

where:

acigroup is an ACIGROUP group name.
access is the authorization level based on the DEDICATE or NICDEF statements. If the DEDICATE statement specifies the R/O operand, access is READ. In all other instances UPDATE access is granted. If the NICDEF statement specifies PROMISCUOUS mode, access is CONTROL. In all other instances UPDATE access is granted.
addr is the virtual address of the minidis.
devno is the real device number specified on the DEDICATE statement.
byuserid is the user id specified on the LOGONBY statement.
mode is the authorization level based on link mode (NONE, READ, UPDATE, CONTROL, or ALTER).
name is the specific name of that Guest LAN or Virtual Switch.
owner is the user id of the owner of the minidisk defined in the LINK statement, or the owner of the Guest LAN or Virtual Switch specified on the NICDEF statement.
password is the password from the user's directory entry.
sysname is the system identifier for the processor on which you run z/VM. For IDENTITY defined users the sysname is taken from the corresponding BUILD statement. For all other users, or when the sysname cannot be determined, a placeholder of "systemid" is used.
userid is the ID of the user whose CP directory entry is being processed.
vlanid is a number 0000-4095 that represents the VLAN to which access is being granted.

See “Access Authority for Minidisks on z/VM” on page 154 to learn how LINK modes are mapped to RACF access levels.

See “Processing OpenExtensions Statements in the CP Directory” on page 49 for information on how RPIDIRCT processes OpenExtensions-related CP directory statements.

The maximum number of users allowed in a group is 5200. If your system has more than 5200 users and the group fills up, you are prompted to supply a new group name.

Consider editing the RPIDIRCT SYSUT1 file to change the UACC from NONE to UPDATE for those readers that any user ID can spool to.

If you would like to keep existing permissions that are in a RACF database, edit the RPIDIRCT SYSUT1 file, and delete the RESET operand from all the PERMIT commands. The PERMITs by default have the option RESET(ALL). RESET(ALL) specifies that RACF is to delete from the profile both the entire current standard access list and the entire current conditional access list. You can delete all the RESETs by XEDITing the RPIDIRCT SYSUT1 file and issuing `CHANGE /RESET// *` and then FILE.

Move your key z/VM users and their minidisks to the front of the file so that RACF defines them first (for example, your own user ID and the operator user IDs). These users can then log on shortly after the build process begins.

Note:

1. The ADDUSER command for a user ID might fail if you have already defined that user ID. This situation can occur if you are sharing your RACF database, and the same user ID is defined on both sharing systems. To prevent the failure, remove or comment out the ADDUSER command by placing an asterisk in column one.
2. The LINK information in a CP directory entry is transferred to the RACF profiles now protecting the various minidisks. If minidisk owners want to change this arrangement, they must wait until after RACF has been installed. After reviewing the access lists for their minidisks, they should decide who should have continued access. If the owners wish to delete users from an access list, they should enter a PERMIT command or use the RACFPERM EXEC.
3. RACF bases the level of access that it grants on the authorization it finds in the directory LINK or MDISK statement. When you specify a read password of "ALL" on the MDISK statement, this causes UACC(READ) to be set. Anything other than a read password of "ALL" causes UACC(NONE) to be set.
4. In order to use the z/VM automated service commands you need to give UACC of UPDATE for VMRDR to the MAINT user ID.

Run RPIDIRCT to Create the RPIDIRCT SYSUT1 File

The RPIDIRCT EXEC needs access to three minidisks:

- A minidisk with the CP directory
- A minidisk with the DirMaint cluster files (if applicable)
- A minidisk for the RACF command output generated by the RPIDIRCT EXEC

1. You should be logged on to the RACF Administrator user ID you have defined, for example, RACFADM. See “Defining Administrator User IDs for Your Own Use” on page 40.
2. Link and access the disk that contains your CP directory file. (The default CP directory file name is USER DIRECT and it resides on MAINT's 2CC minidisk.)

   Note: If you are currently running DirMaint then you need to issue the DIRM USER WITHPASS command and put the resulting file on MAINT's 191 A-disk and call it USER WITHPASS. This is the name of the file you want to use in the step that follows that has you issue the rpidirct command. Also you need to link MAINT's 191 disk instead of the 2CC disk.
3. Consider issuing

   ```
   DIRECTXA filename filetype filemode (EDIT
   ```

   to check the validity of your CP directory.
4. If applicable, link and access the disk that contains DirMaint cluster files. By default the DirMaint cluster files reside on the DIRMAINT 1DF disk. You will need to know the READ password in order to link to this disk.
5. The RPIDIRCT exec currently expects the cluster files to be located on a disk that has been accessed with a file mode of "G". This file mode is hardcoded in the RPIDIRCT exec. Ensure that the cluster files disk is accessed with a file mode of "G".
6. Ensure that the RACF Administrator user ID 191 disk has enough free space and that the user ID building the database (later in the installation) has access to the RACF Administrator user ID 191 disk.
7. Make sure the CP directory entries will not cause any problems. See “General CP Directory Requirements” on page 44 for the problem areas you should check.
8. Create the CMS file of RACF commands, RPIDIRCT SYSUT1. This file will be used by another step further on in the install process.

Processing OpenExtensions Statements in the CP Directory

z/VM includes a set of CP directory control statements that are used to define OpenExtensions information to z/VM. By default, these are processed along with the other supported directory

statements. In addition, you can use the OVM option of the RPIDIRCT EXEC to process only the OpenExtensions information in the CP directory.

Note: It is recommended that you run the DIRPOSIX utility *before* you run the RPIDIRCT EXEC. DIRPOSIX, a sample utility provided by z/VM, assigns a unique UID to each user defined in the CP directory. See *z/VM: CP Planning and Administration* for more information.

DIRPOSIX may also create new users and groups that must be changed. See "DIRPOSIX Considerations" on page 51 for more information.

You can use the OVM option of the RPIDIRCT EXEC to process only the OpenExtensions information in the CP directory, as follows:

```
RPIDIRCT filename filetype filemode outmode (OVM
```

When you are satisfied with the output from RPIDIRCT, invoke RPIBLDDS as follows:

```
RPIBLDDS filename
```

where *filename* is the name of the SYSUT1 file created by RPIDIRCT.

If the OVM option is not specified, all of the RACF-related statements are processed, including the OpenExtensions statements.

Note: This step might take a while to run, and the output comes to the console. If you do not want to see the output then spool your console before you enter the command (so that you will be able to view any error messages) and type HT after the RPIDIRCT command starts.

Table 8 on page 50 shows which RACF statements are generated for each of the OpenExtensions statements in the CP directory. See *z/VM: CP Planning and Administration* for descriptions of the CP directory control statements.

Table 8. RPIDIRCT Processing for OpenExtensions

For these CP directory statements:	**These RACF commands are generated:**
The first occurrence of: POSIXOPT QUERYDB in the CP directory	RDEFINE VMPOSIX POSIXOPT.QUERYDB UACC(READ)
POSIXOPT QUERYDB DISALLOW	PERMIT POSIXOPT.QUERYDB CLASS(VMPOSIX) ID(*userid*) ACCESS(NONE)
The first occurrence of: POSIXOPT SETIDs in the CP directory	RDEFINE VMPOSIX POSIXOPT.SETIDS UACC(NONE)
POSIXOPT SETIDs ALLOW	PERMIT POSIXOPT.SETIDS CLASS(VMPOSIX) ID(*userid*) ACCESS(READ)
POSIXINFO UID *uid*	ALTUSER user OVM(UID(*uid*))
POSIXINFO GNAME *gname*	CONNECT user GROUP(*gname*)
POSIXINFO IWDIR *string*	ALTUSER user OVM(HOME(*string*))
POSIXINFO IUPGM *string*	ALTUSER user OVM(PROGRAM(*string*))
POSIXINFO FSROOT *string*	ALTUSER *userid* OVM(FSROOT(*string*))
POSIXGROUP *gname gid*	ADDGROUP *gname* ALTGROUP *gname* OVM(GID(*gid*))
POSIXGLIST GNAMES *gname-1...gname-n*	CONNECT *userid* GROUP(*gname-1*) ⋮ CONNECT *userid* GROUP(*gname-n*)

Note:

1. When the GIDS parameter is used on the POSIXGLIST statement, or the GID parameter is used on the POSIXINFO statement, the GID is changed to its corresponding group name (*gname*) for use in the RACF CONNECT command.
2. RACF does not allow a profile in the GROUP class to have the same name as a profile in the USER class. Thus, you cannot have a group name on a POSIXGROUP statement that is the same as any user ID on a USER statement. (RACF first converts the group name to uppercase.) RPIDIRCT does not issue any error messages in this situation, but, when the RPIBLDDS EXEC executes the commands generated by RPIDIRCT, either the ADDUSER or the ADDGROUP will fail, depending on the order of execution.
3. For each *gname* on a POSIXGROUP statement, RPIDIRCT first generates an ADDGROUP statement to define the group, and then an ALTGROUP statement to add the GID to the group. It is done this way in case you intentionally used existing RACF group names on POSIXGROUP statements. If a single ADDGROUP command had been generated, it would fail, and a GID would not be assigned to the group. If you did use existing RACF group names, you can choose to remove the ADDGROUP commands from the output file, or simply let them run and they will fail without any harm.
4. POSIXOPT EXEC_SETIDS statements will not result in the creation of any RACF commands to create or modify EXEC.U*uid* or EXEC.G*gid.* profiles in the VMPOSIX class. There is no one-to-one correspondence between the CP directory and RACF support of executing set-UID and set-GID files in the byte file system, and careful consideration should be given in implementing these controls. See “Protecting Set-UID and Set-GID Executable Files” on page 205 for details on protecting set-UID and set-GID files.
5. A user's default RACF group is not changed as a result of any POSIXINFO or POSIXGLIST statements.
6. OpenExtensions defaults established in the system configuration file are not processed by RPIDIRCT.
7. If you used DIRPOSIX with the SYSENTRIES option, your CP directory will contain a reference to the implicitly-defined group called DEFAULT. RPIDIRCT will issue a message referring to this undefined group. This is expected and can be ignored.

DIRPOSIX Considerations

In addition to adding UID values for all defined users in your CP directory, DIRPOSIX also defines users and groups that are common on many UNIX systems. However, RACF cannot support all the entries as created by the SYSENTRIES option for the following reasons:

- Certain UID and GID values will not be valid on RACF ADDUSER and ALTUSER commands (for example: 4294967294 and 4294967295).
- Some group names are identical to user names (`bin` and `sys`, for example). RACF does not allow both a user profile and a group profile with the same name.

RPIDIRCT allows for these differences by providing a control file, RPIDIRCT CNTRL, which contains a set of UID, GID, and group name values that are acceptable to RACF. RPIDIRCT substitutes these values for the SYSENTRIES default values in the output file created by RPIDIRCT. Some of the names are referenced in other OpenExtensions configuration files.

The BFS LOADBFS configuration file is used to install the byte file system. The file is used once during installation and is not referenced again. See *z/VM CMS File Pool Planning, Administration, and Operation* for more information.

The SHELL LOADBFS configuration file is used to install the optional Shell and Utilities feature, and is also used every time the Shell and Utilities are serviced. This file is owned by the VMSES/E user ID that is used to install the Shell and Utilities feature. Both of these files refer to users `bin` and `root` and groups `bin` and `system`. If you implement RACF protection of OpenExtensions *after* OpenExtensions for z/VM has been installed, file ownership is already established in the byte file system. It is best not to change this ownership. Within the file system, ownership is based on the UID and GID values, not the user name and group name values.

When you implement RACF protection, it is recommended that you do the following:

1. Run the DIRPOSIX utility with the SYSENTRIES option.

2. Run the RPIDIRCT utility to automatically change the UID and GID values according to the options set in the RPIDIRCT CNTRL file.

   Any conflicting group names are changed to the group names shown in Table 9 on page 52.

   RPIDIRCT leaves any GID or UID of 4294967295 as undefined. By convention, if a group or user is defined but does not have an OVM segment, RACF assumes a GID or UID of 4294967295.

   See Table 9 on page 52 for the changes made by RPIDIRCT. You can modify RPIDIRCT CNTRL if the names are not acceptable at your installation. The syntax of the control statements is documented in the file. The supplied RPIDIRCT CNTRL file is located on the PMAINT 551 disk.
3. Replace all occurrences of the group `bin` in BFS LOADBFS (and in SHELL LOADBFS if you have installed the optional OpenExtensions Shell & Utilities Feature) with `GBIN`.

   If you have modified RPIDIRCT CNTRL to use a value other than `GBIN`, you must make the equivalent change in BFS LOADBFS (and SHELL LOADBFS).

Table 9 on page 52 maps the values added to the CP directory by the DIRPOSIX SYSENTRIES option to values that are compatible with RACF.

Table 9. RPIDIRCT CNTRL Changes to DIRPOSIX SYSENTRIES

DIRPOSIX Default Value	**RACF-Compatible Value**
system	SYSTEM
staff	STAFF
bin	GBIN
sys	GSYS
adm	GADM
mail	MAIL
security	SECURITY
nobody	GNOBODY
4294967294	2147483647

Chapter 4. Defining Users

This chapter provides in-depth information on defining users for z/VM systems.

All users defined to CP must also be defined to RACF. Users who are not defined to RACF cannot use the system or access any of its resources. It is possible to define users to RACF who are not defined to CP in the user directory. While they do not have a virtual machine of their own, these users can access system resources from another system using FTP, APPC, or other remote resource access protocols. When authorized, they can also logon to other users using LOGON BY.

Note: See Chapter 12, “Controlling OpenExtensions and BFS Security,” on page 195 for information about RACF security at the Portable Operating System Interface (POSIX) 1003.1 level.

User Profiles

When you define a user to RACF, you create a user profile in the RACF database. A user profile can consist of a RACF segment, and optionally, an OVM segment.

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. You can also list the contents of an entire user profile, or the contents of individual segments of the user profile using the LISTUSER command. For information on how to use these commands, see *z/VM: RACF Security Server Command Language Reference*.

Overview of Authentication

When a user logs on to a z/VM system, he or she must supply an authenticator to prove their identity. RACF has several methods available to authenticate the user ID, and part of the planning process is deciding which method(s) are right for the different user IDs on the system. These methods include:

Password
A password is a traditional 1- to 8-character alphanumeric value. The simplicity of passwords can present a relatively easy point of attack for exploitation. In order for systems that rely on passwords to be secure, they must enforce password controls and provide user education. Some of the common problems with having a simple password are that users tend to choose common passwords, write down their passwords, and/or unintentionally install malware that can log and forward passwords to the attacker.

Password Phrase
A password phrase (passphrase) is a character string consisting of mixed-case letters, numbers, and special characters including blanks. Password phrases have security advantages over passwords in that they are long enough to withstand most hacking attempts yet are unlikely to be written down because they are easy to remember.

Multi-Factor Authentication (MFA)
A more secure option than the common password or password phrase authentication is for systems to require multiple authentication factors to verify the user's identity. A multi-factor authentication system requires that multiple authentication factors be presented during logon to verify a user's identity. Each authentication factor must be from a separate category of credential types:

- Something you know – a password or security question
- Something you have – an ID badge or cryptographic token device
- Something you are – a fingerprint or other biometric data.

By requiring multiple authentication factors, a user's account can not be compromised if one of their factors is discovered. IBM Multi-Factor Authentication for z/VM provides support for authenticating

with multiple authentication factors. RACF users can be configured to require authentication through MFA. For these select users, RACF calls MFA to help in making the authentication decision during log on processing.

Recommendations for Human Users

For human users, the choice of the type of authenticator(s) to assign will depend on your policy, and on which applications are used by the user:

- If you have many customers/clients or general employees logging on, you do not want their passwords/passphrases written down. Multi-Factor Authentication (MFA) with the NOPWFALLBACK option allows these users to logon with the most secure method. For more information on MFA, see "Multi-Factor Authorization" on page 74.
- If the user authenticates to z/VM using an application that does not support password phrases or MFA, the user must be assigned a password. In this case, if the user also logs on directly to CP, the user can also be assigned a password phrase. For more information, see "Passwords and Password Phrases" on page 67 and Chapter 19, "Using the Secured Signon Function," on page 283.

Recommendations for Service Machines and Linux Guests

Disconnected service machines and multi-user operating systems like Linux on IBM Z® should be created as protected users. Protected users do not have a password, password phrase, or any other enabled authentication method and can only be started by using XAUTOLOG or LOGONBY. Any direct human access to these guest's terminals should be authenticated and authorized using LOGONBY and human user IDs, creating a personalized audit trail. For more information, see "Defining Shared User IDs" on page 79.

Recommendations for MFA Authentication

A limited set of employees who maintain the system and administrate RACF for VM should be permitted to LOGON with FALLBACK for circumstances when MFA might be unavailable, unresponsive or configured incorrectly. PWFALLBACK allows these user IDs to logon with a password or passphrase independent from the availability of MFA with LOGON FALLBACK.

Because MFA is often not a good fit for IDs that are used for automation purposes such as automatic data transfer via FTP, it is recommended that you use strong password phrases to authenticate these user IDs if direct access is necessary.

Sufficient time should be allocated at the initial implementation of the MFA authentication policy for training and preparation. Users should be informed about the different characteristics of the implemented policy and factors and the authentication limit and validity period of the derived MFA credential.

For more information on MFA, see "Multi-Factor Authorization" on page 74.

The RACF Segment in User Profiles

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

USERID
User ID of the user

NAME
User's name

OWNER
Owner of the user's profile

DFLTGRP
Default group for the user

AUTHORITY
User's authority in the default group

PASSWORD
User's password

NOPASSWORD
Indicates that the user cannot enter the system using a password and, when the user also has the NOPHRASE attribute and is not enabled for MFA, gives the user the PROTECTED attribute.

PHRASE
User's password phrase

NOPHRASE
Indicates that the user cannot enter the system using a password phrase and, when the user also has the NOPASSWORD attribute and is not enabled for MFA, gives the user the PROTECTED attribute.

MFA
Indicates that the user is enabled for MFA (Multi-Factor Authentication). The user will have the PROTECTED attribute if the user is not enabled for MFA and has the NOPHRASE and NOPASSWORD attribute.

PWFALLBACK
Indicates that the MFA enabled user is allowed to use the LOGON FALLBACK parameter and log on using their password/passphrase.

REVOKE
Date when RACF prevents the user from gaining access to the system

RESUME
Date when RACF gives the user access to the system again

UACC
Default universal access authority for resources the user defines

WHEN
Days of the week and/or hours of the day the user can have 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 responsibility for auditing system resources

ROAUDIT
Gives the user the system-wide ROAUDIT attribute

OPERATIONS
Gives the user the system-wide OPERATIONS attribute

DATA
Installation-defined data

SECLABEL
User's default security label

The OVM Segment in User Profiles

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

UID
User's OpenExtensions VM user identifier

HOME
User's OpenExtensions VM initial directory path name

PROGRAM
User's OpenExtensions VM program path name, such as a default shell program

FSROOT
User's OpenExtensions VM file system root directory

See "Defining OpenExtensions Users" on page 198 for more information.

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 by way of 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 by way of field-level access control. See "Setting Up Field-Level Access for the OVM Segment" on page 203 for more information.

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')
- The characters, #, $, and @, may be displayed differently on terminals outside the United States; therefore, use the characters with the hexadecimal equivalents shown above. Note that this is more restrictive than that allowed by CP in the user directory. Specifically, RACF does not allow the underscore or hyphen (dash). It is recommended that you not define user IDs that contain #, $, or @. These characters may not display the same way on all terminals, and # and @ are default line editing symbols.
- No two user IDs can be the same. No user ID can be the same as a group name.
- With OpenExtensions, the user identifier (UID) is an integer value that defines a user. Although you can use the same integer value to define multiple users, it is not recommended. If you use the same integer value:
  - Control at an individual user level is lost because the UID is used in OpenExtensions security checks.
  - Users with the same UID value are treated as a single user during OpenExtensions security checks.

  A UID of 0 is used to define an OpenExtensions superuser. A superuser can perform any OpenExtensions function and passes all OpenExtensions security checks.

Suggestions for Defining User IDs

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 group RESEARCH. 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 when many of the current users will have changed groups. 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.

Migrating Existing User IDs to RACF

On z/VM, the installation procedure executes several EXECs that help in migrating users to RACF. For more information, see "Using RPIDIRCT to Prime the RACF Database from the CP Directory" on page 44 and *z/VM: RACF Program Directory*.

Note: The z/VM directory allows user IDs to have special characters such as the hyphen (-) that are not allowed in RACF. If you use the RPIDIRCT EXEC to convert z/VM directory statements into RACF commands, the RPIDIRCT EXEC changes the hyphens into dollar signs ($).

Creating New User IDs from Scratch

Where user IDs are being assigned from scratch, they can often be created in blocks through a user-written EXEC on z/VM. For example, you could centrally create 50 user IDs, MKT01 through MKT50, and allocate them to the manager of group MKT to assign to the 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.

Ownership of a RACF User Profile (including the ability to modify the password or prevent the user from entering the system)

Each user defined to RACF has a user profile; 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 system-SPECIAL) can modify, list, and delete the user's profile and 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 39 on page 221.

User Attributes

User attributes are extraordinary capabilities, restrictions, 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 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 indicated in the user's profile, and are in effect regardless of the group the user is connected to.

The following sections describe these user attributes:

- SPECIAL
- AUDITOR
- ROAUDIT
- OPERATIONS
- CLAUTH
- REVOKE
- PROTECTED

SPECIAL Attribute

A user with the SPECIAL attribute at the system level can issue all RACF commands. The SPECIAL attribute gives the user full control over all 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. Personnel having the SPECIAL attribute should be required to use passwords, and should change their passwords often to help ensure security.

You can assign the SPECIAL attribute at the group level. When you do, the *group-SPECIAL* user has full control over all profiles within the scope of the group. See “User Attributes at the Group Level” on page 61 for additional details.

For a list of the RACF commands that this attribute allows users to issue, see Table 32 on page 216.

AUDITOR Attribute

A user with the AUDITOR attribute at the system level has the authority to specify logging options on the:

- ALTUSER, RALTER, and SETROPTS commands
- ALTDIR and ALTFILE commands
- ALTDSD command

In addition, the auditor can list auditing information with the:

- LISTGRP, LISTUSER, RLIST, and SEARCH commands
- IRRUT100 utility program
- LDIRECT, LFILE, SRDIR, and SRFILE commands
- LISTDSD command

The AUDITOR attribute gives the auditor control of logging to the SMF data file. 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 having the AUDITOR attribute can list all profile information available to the SPECIAL user, as well as information that is available to auditors. Note, however, that this extended listing capability does not give the auditor any additional authority to change information in the RACF database; nor does it give additional access to protected data.

A user must have the AUDITOR or ROAUDIT attribute 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.

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. See "User Attributes at the Group Level" on page 61 for detailed information.

For a list of the RACF commands that this attribute allows users to issue, see Table 33 on page 218.

ROAUDIT Attribute

A user who has the ROAUDIT attribute has the authority to list auditing information using the LISTDSD, RLIST, LISTUSER, LISTGRP, SETROPTS LIST, and SEARCH commands, as well as the IRRUT100 utility. Unlike users with the AUDITOR attribute, users with the ROAUDIT attribute are unable to specify logging options or to control logging to the SMF data file.

The user who has the ROAUDIT 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.

A user must have the AUDITOR or ROAUDIT attribute to run the DSMON program.

You should assign the ROAUDIT attribute only to users who are responsible for auditing RACF security controls and functions, but who are not to be responsible for establishing those security controls or functions. For example, you might give the ROAUDIT attribute to a user account created for an external auditor to permit that person to audit the security controls and function without being able to alter those controls. To provide a check and balance on RACF security measures, you should give the ROAUDIT attribute to auditors or users other than those who have the SPECIAL or AUDITOR attribute.

The ROAUDIT 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 ROAUDIT attribute should take special care to protect their own data files, because they can contain sensitive information.

For a list of the RACF commands that this attribute allows users to issue, see Table 34 on page 219.

OPERATIONS Attribute

A user with the system-OPERATIONS attribute has full authorization to all RACF-protected resources in the DASDVOL, DATASET, DIRECTRY, FILE, GDASDVOL, PSFMPL, RODMMGR, 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 system-wide OPERATIONS users members of any group that is in the access lists of resource profiles.
- Security classification checking or security label checking can deny access.

You can limit the access allowed because of the OPERATIONS attribute in two ways:

- By placing the OPERATIONS user or a group the user is connected to in the access list of sensitive resources using the PERMIT, PERMDIR, or PERMFILE 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

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, have a user with the system-AUDITOR attribute issue the following command:

```
SETROPTS OPERAUDIT
```

To reduce the number of users having the OPERATIONS attribute at the system level (and therefore having the attribute for all resources on the system, you can assign the OPERATIONS attribute at the group level. When you do, the *group-OPERATIONS* user's authority is restricted to resources within the scope of the group. For more information, see “Scope of Authority for group-SPECIAL, group-AUDITOR, and group-OPERATIONS Users” on page 61 and “User Attributes at the Group Level” on page 61.

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 35 on page 219.

CLAUTH (Class Authority) Attribute

Users receive the CLAUTH attribute on a *class-by-class* basis. 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), RACF allows the user to define profiles in that class.

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

Note:

1. CLAUTH has no meaning for the FILE and DIRECTRY classes.
2. The authority of all users to define profiles in general resource classes can be restricted by issuing the SETROPTS GENERICOWNER command. See “Restricting the Creation of General Resource Profiles (GENERICOWNER Option)” on page 103.
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 IBM-supplied classes, see the description of the class descriptor table in *z/VM: RACF Security Server 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, provided 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 system-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 36 on page 219.

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 with the SPECIAL attribute) can assign the REVOKE attribute.

PROTECTED Attribute

You can define a protected user ID by assigning the NOPASSWORD, NOPHRASE, and NOMFA 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 inactivity or unsuccessful attempts to access the system using incorrect passwords, password phrases, and MFA credentials. However, they can be revoked using the ALTUSER (*userid*) REVOKE command. 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 CP logon, rlogin, or FTP. 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 or password phrase is required.

You might wish to assign protected user IDs to disconnected service machines, maintenance user IDs, and Linux image user IDs, to minimize their exposure to inadvertent or malicious misuse or revocation. These are the same types of user IDs that you should consider defining as shared user IDs.

The following example shows the ALTUSER command used to assign the PROTECTED attribute to an existing user ID.

```
ALTUSER SERVER8 NOPASSWORD NOPHRASE NOMFA
```

Note that any user ID is by default a protected user ID unless a password or password phrase is assigned, or the user is enabled for MFA. For example, the following command creates a protected user ID.

```
ADDUSER LINUX1
```

A protected user ID will have the PROTECTED attribute displayed in the output of the LISTUSER command.

To remove the PROTECTED attribute from an existing user ID, use the ALTUSER command to assign a password or password phrase, or enable the user for MFA:

```
ALTUSER SERVER8 PASSWORD(password)
```

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 “List-of-Groups Authority Checking” on page 94.

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.

Scope of Authority for group-SPECIAL, group-AUDITOR, and group-OPERATIONS Users

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 (refer to Figure 4 on page 64 and Figure 5 on page 65):

- Resources owned by the group (for example, GROUP1.DATA owned by GROUP1)
- Resources owned by users who are owned by the group (for example, USER2.DATA owned by USER2 who is owned by GROUP1)
- Resources owned by subgroups that are owned by the group (for example, GROUP2.DATA owned by GROUP2 that is owned by GROUP1)
- Resources owned by subgroups that are owned by subgroups, owned by the group, and so on (for example, GROUPZ.DATA owned by GROUPZ that is owned by GROUP2 that is owned by GROUP1).

Note that the scope of the group does *not* extend to the following resources:

- Resources owned by groups owned by users who are owned by the group (for example, GROUPY.DATA owned by GROUPY 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).

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 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/VM: RACF Security Server Auditor's Guide*.

The limits of the security administrator, group administrator, auditor, and operations personnel authority at the group level are described in Table 10 on page 62. (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 10. Scope of Authority for User Attributes at the Group Level

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 owned by the group, by a subgroup 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 with the group-AUDITOR attribute can perform all of the functions of an auditor, but is restricted to the same subset of users as the user with the group-SPECIAL attribute.
Groups	**Group-SPECIAL attribute:** A user with 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.
SFS files and directories	**Group-SPECIAL attribute:** A user with the group-SPECIAL attribute has full authority to work with FILE and DIRECTRY profiles: • Owned by the group • Owned by users or groups that the group owns • With a second qualifier that is a user identifier owned by the group. The group-SPECIAL user can also define FILE and DIRECTRY profiles with a second qualifier that is a user 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 restricted to the same subset of SFS files and directories as the user with the group-SPECIAL attribute.

Table 10. Scope of Authority for User Attributes at the Group Level (continued)	
Resource	**Attribute, User, and Authority**
General resources	**Group-SPECIAL attribute:** A user with 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 with the AUDITOR or OPERATIONS attribute can perform all of the functions of an auditor or operator, but is restricted to the same above subset of resources as the user with the group-SPECIAL attribute.

The following two figures show the scope of authority of a group-SPECIAL user. Figure 4 on page 64 shows a typical authority structure containing three major groups: Group 1, Group 2, and Group 3.

Note: Although this figure uses z/OS terminology to illustrate relationships between users and groups and their resources, the same concepts apply to relationships between users and groups and their resources on z/VM systems.

Figure 5 on page 65 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 nonshaded area) in part 2 of this figure.

USER1 has authority to the profiles in the nonshaded area for the reasons listed in Table 10 on page 62. 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. If USER4.DATA is a z/OS data set, USER1 cannot display the profile information for this data set with LISTDSD, even if USER2, for example, is in its access list. (However, by using IRRUT100, RACF informs USER1 that USER2 is in the access list of USER4.DATA.)
- U4A is not a general resource owned by a user who is owned by GROUP1.

Note: Although this figure uses z/OS terminology to illustrate relationships between users and groups and their resources, the same concepts apply to relationships between users and groups and their resources on z/VM systems.

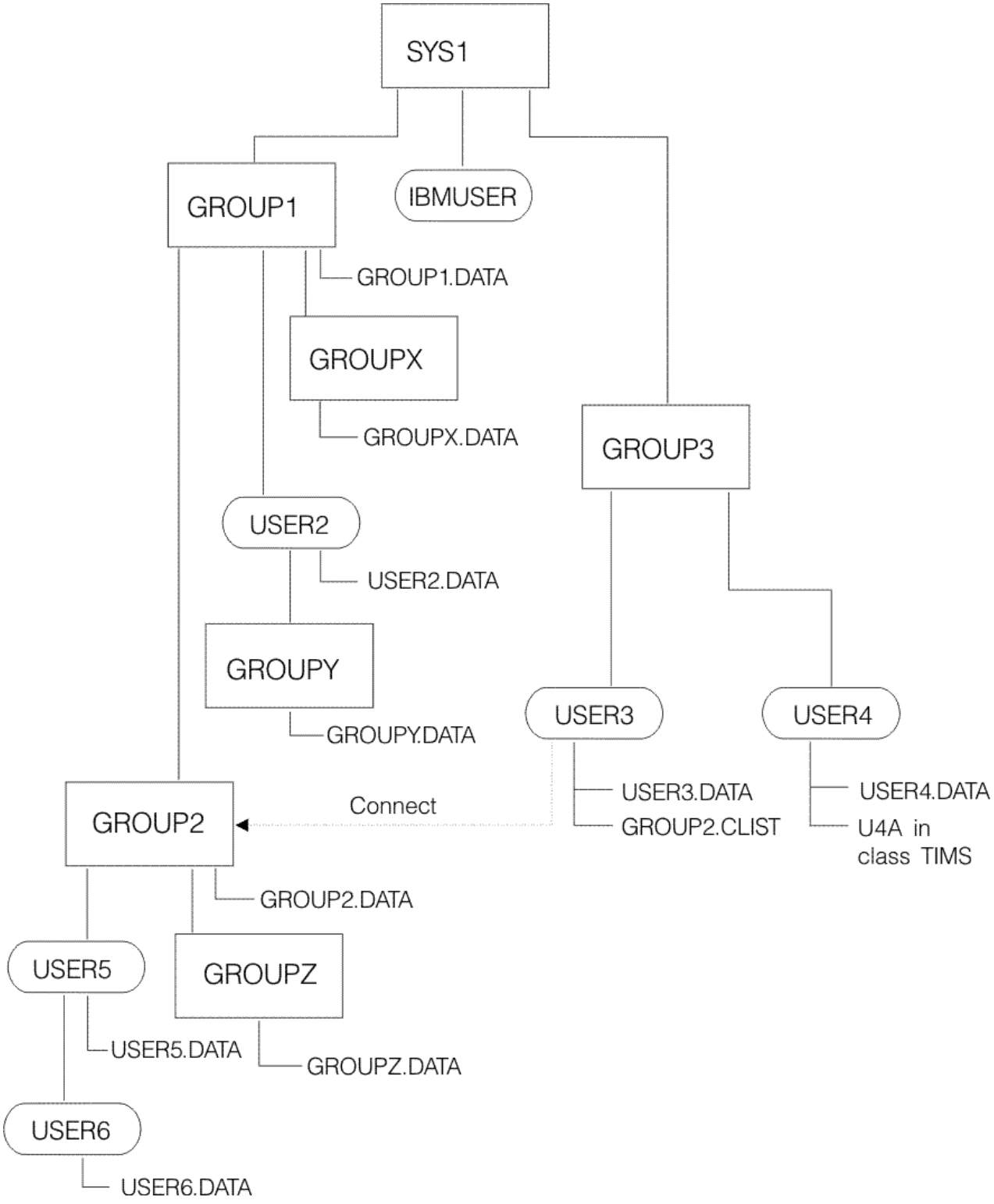


Figure 4. Group-Level Authority Structure

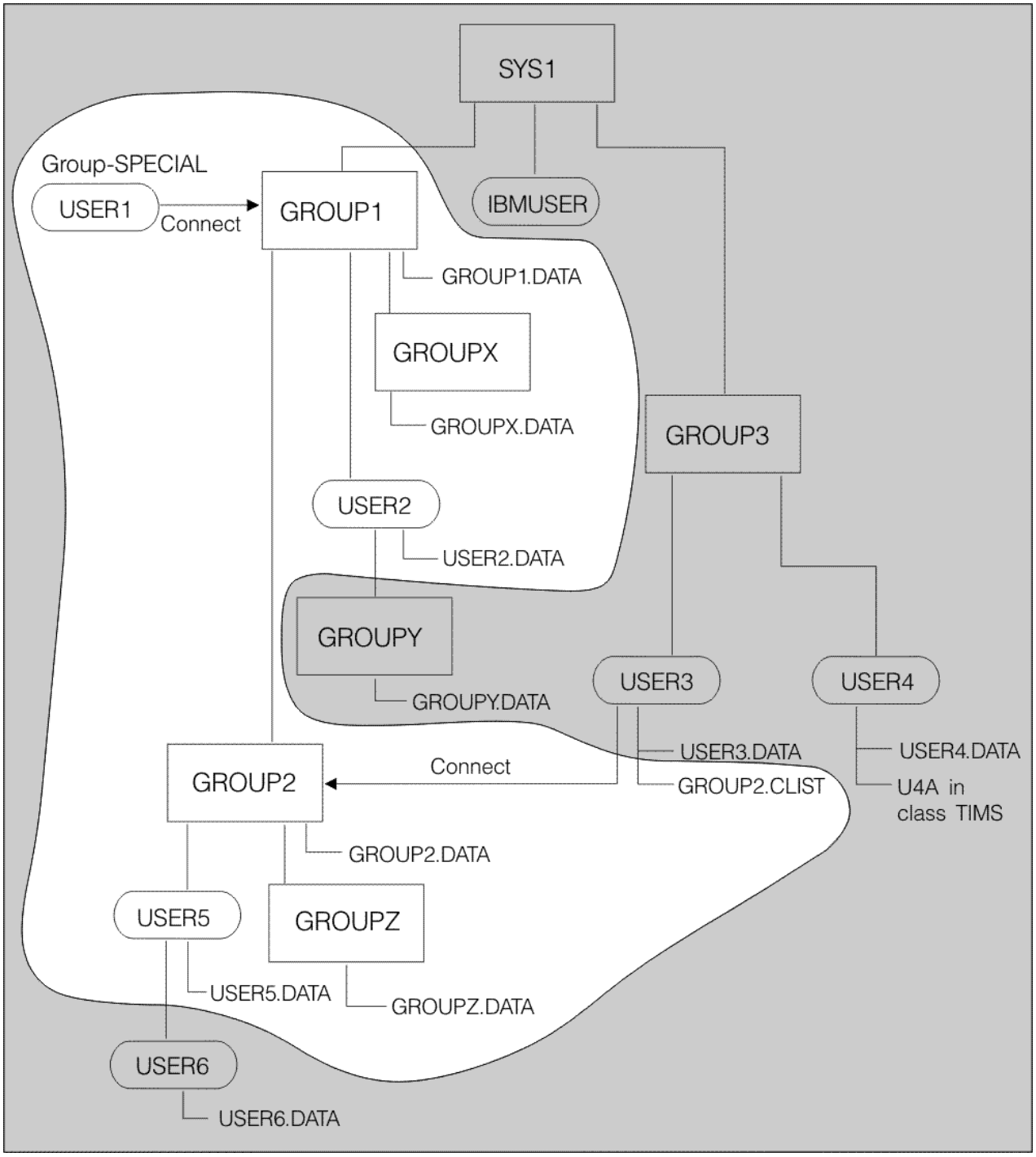


Figure 5. Scope of Authority of a Group-SPECIAL User

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, ROAUDIT, and OPERATIONS attributes to only those users responsible for administering RACF on a system-wide basis
- CLAUTH attributes to only those users who will define other users and general resources (other than SFS files and directories).

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, ROAUDIT, 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, ROAUDIT, and REVOKE attributes, both at 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 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 a user defines while connected to the specified default group. When a user issues the ADDDIR, ADDFILE, or 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.

For more information on using the UACC operand on the ADDUSER, ALTUSER, or CONNECT command, see *z/VM: RACF Security Server Command Language Reference*.

Assigning Security Categories, Labels, and Levels 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 a 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, then 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 specified in the resource profile. Instead, RACF performs security label authorization checking, which involves the security levels and categories 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 313.)

For more information on setting up and using security classification of users and data, see Chapter 17, "Security Classification and Zoning," on page 249.

Passwords and Password Phrases

When a user logs on to a z/VM system, he must supply an authenticator to prove he is who he says he is. That authenticator can be either a password or a password phrase. A password is a traditional 1 to 8 character alphanumeric value. A password phrase is a character string consisting of mixed-case letters, numbers, and special characters including blanks. Password phrases have security advantages over passwords in that they are long enough to withstand most hacking attempts yet are unlikely to be written down because they are so easy to remember. A user can be assigned a password, a password phrase, or both.

When a user profile is created, there is no default value assigned to either the password or password phrase, and the user is not enabled for MFA. The user is a protected user and will not be able to logon. This is a perfectly acceptable, and even preferable, situation for a disconnected service machine or Linux image user ID. See "PROTECTED Attribute" on page 60 for more information.

For humans, the choice of the type of authenticator(s) to assign will depend on your policy, and on which applications are used by the user. If the user authenticates to z/VM using an application that does not support password phrases, then the user must be assigned a password. In this case, if the user also logs on directly to CP, then the user may also be assigned a password phrase.

A user will not be able to assign himself an initial password or password phrase. When you do assign one, the user can change that value at any time, but will not be able to remove it. When assigning an initial value, be sure it is difficult to guess. By default, the user will be forced to change this initial value the first time it is used.

To assign a password, use the PASSWORD operand of the ALTUSER command:

```
ALTUSER GLENN PASSWORD(g1GgiTty)
```

Or, to remove a password:

```
ALTUSER GLENN NOPASSWORD
```

To assign a password phrase, use the PHRASE operand of the ALTUSER command:

```
ALTUSER STEWIE PHRASE('I shall rule the world!')
```

Or, to remove a password phrase:

```
ALTUSER STEWIE NOPHRASE
```

These sample commands will assign a value that must be changed by the user when he first logs on, thus insuring that from that point on, the user is the only one who knows his password. Note that there is a NOEXPIRED option of the ALTUSER command, which assigns a password or phrase that does not need to be changed when the user logs on. This is intended for use by trusted applications that set RACF passwords on behalf of users (e.g. a password synchronization application). It should not be used by administrators, because the principle of user accountability rests on the idea that a user is the only one who knows his own password. See *z/VM: RACF Security Server Command Language Reference*

Notes:

1. In general, the password in a user's CP directory entry is ignored when RACF is active. However, there are a small number of special values that have meaning even when RACF is active. See *z/VM: CP Planning and Administration* for details.

   NOLOG - The user cannot enter the system.

   AUTOONLY - The user can only be XAUTOLOGed. This is the same as defining the user with NOPASSWORD and NOPHRASE attributes.

   NOPASS - The user can logon without specifying a password. The use of this very sensitive option should be carefully controlled by the security administrator. It should only be used in cases where the user ID has access only to non-sensitive information that you intend to be viewed anonymously. Its use is controlled by means of the IRR.NOPASS profile in the FACILITY class. If the profile is not defined, or the FACILITY class is not active, any NOPASS user can logon without specifying a password. If the profile is defined, and the FACILITY class is active, NOPASS users can only logon if they have been granted READ access to the profile. IBM recommends that the IRR.NOPASS profile be created with UACC(NONE).
2. Passwords are one-way encrypted in the RACF database by default. However, RACF can be configured to envelope passwords such that they are recoverable in clear text by trusted applications, such as a password synchronization application. See Chapter 16, “Password and Password Phrase Enveloping,” on page 237 for details.

The following topics describe various considerations and options for passwords and password phrases.

Password Syntax Rules

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:

- Minimum and maximum length of passwords
- Character content of installation-selected positions in the passwords

Restriction: The password syntax rules you define are not enforced when users log on with their current passwords. Therefore, changes you make to your password syntax rules will not affect users with current passwords. Your changes will take effect for current users only when they change their passwords. For new users, the changes will take effect when the new user logs on for the first time. In addition, password syntax rules are not enforced when you define a temporary password for another user using the ALTUSER PASSWORD command unless you specify the NOEXPIRED option.

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 and contain vowels in positions 1, 3, 5, 6, 7, and 8 and numbers in positions 2 and 4. Thus, the password “A2E2EAEE” follows the rule, and “C3DMIER5” does not.

The following example shows a command establishing syntax rule RULE2.

```
SETROPTS PASSWORD(RULE2(LENGTH(8) MIXEDALL(1:8)))
```

Syntax rule RULE2 specifies that new passwords must be 8 characters in length and contain at least one character from every active character category somewhere in the password. Up to four character categories can be active:

- Uppercase alphabetic characters (not including the national characters)
- Lowercase alphabetic characters, if SETROPTS PASSWORD(MIXEDCASE) is enabled
- Numeric characters
- National characters, and the special characters if SETROPTS PASSWORD(SPECIALCHARS) is enabled

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.

Allowing Mixed-Case Passwords

If you have the SPECIAL attribute, you can allow mixed-case passwords for all users on all applications on this system and on all systems that share the RACF database. Use the SETROPTS PASSWORD(MIXEDCASE) option to allow mixed-case passwords at your installation.

```
SETROPTS PASSWORD(MIXEDCASE)
```

Restriction: The ISPF panels do not support the SETROPTS option to activate and deactivate mixed-case password support. For this, you must use the SETROPTS command with the PASSWORD option.

By default, NOMIXEDCASE is in effect and mixed-case passwords are not supported. If you want to allow mixed-case passwords, be sure that mixed-case content is permitted by your password syntax rules. When SETROPTS PASSWORD(MIXEDCASE) is in effect, the RACF commands ALTUSER, ADDUSER, and PASSWORD no longer translate passwords to upper case, nor do applications that provide mixed-case password support.

User considerations: When you activate the MIXEDCASE option, users should be aware of the following considerations.

- Mixed-case passwords are more secure and harder to guess than uppercase passwords. Users are encouraged to select mixed-case passwords.
- Users with existing, uppercase passwords need *not* supply their passwords in upper case. However, once the MIXEDCASE option is activated, any password that is set or changed to a value containing a lowercase character must thereafter be supplied exactly as it was created. In other words, the user must then supply every character of the password using exactly the same case used when the password was created.
- Users are prevented from entering new passwords that differ from their current passwords by only the case in which they are entered. For example, if a user's current password is `IM4JUVE`, the user cannot change it to a new password of `Im4Juve`.

Migration Considerations for Mixed-Case Passwords

Carefully plan your application updates and password rule changes before activating MIXEDCASE. Once MIXEDCASE is activated, subsequently issuing the SETROPTS PASSWORD(NOMIXEDCASE) command may cause unintended results. When you reset to NOMIXEDCASE, users who have mixed-case or lowercase passwords will be unable to enter the system until you reset their passwords.

If you share the RACF database with downlevel systems that do not support mixed-case RACF passwords, or you use a mix of applications that do and do not support mixed-case passwords, do not activate the SETROPTS PASSWORD(MIXEDCASE) option.

Allowing Special Characters in Passwords

If you have the SPECIAL attribute, you can allow special characters to be used in passwords for all users on all applications on this system and on all systems that share the RACF database. Use the SETROPTS PASSWORD(SPECIALCHARS) option to allow special characters in passwords at your installation.

```
SETROPTS PASSWORD(SPECIALCHARS)
```

Restriction: The ISPF panels do not support the SETROPTS option to activate and deactivate support for special characters in passwords. For this, you must use the SETROPTS command with the PASSWORD option. Also, it is recommended that the ISPF panels not be used to enter passwords or password phrases that contain special characters. The "&" and "=" characters have special meaning to ISPF, and using these special characters in passwords or password phrases can cause unexpected results.

Enabling special characters allows the following characters to be specified in RACF passwords.

Hexadecimal value	Symbol (using the EBCDIC 1047 code page)
4B	.
4C	<
4E	+
4F	|
50	&
5A	!
5C	*
60	-
6C	%
6D	_
6E	>
6F	?
7A	:
7E	=

By default, NOSPECIALCHARS is in effect and special characters are not supported. If you want to allow special characters, be sure that they are permitted by your password syntax rules. Syntax rules can be created to require special characters.

The new password exit (ICHPWX01) can be used to further restrict this set when you have characters that are known to present problems with applications that you use.

User considerations: When you activate the SPECIALCHARS option, users should be aware of the following considerations.

- Special character passwords are more secure and harder to guess than uppercase passwords. Users are encouraged to select special characters.
- Certain characters might pose problems for certain applications. Avoid using such characters when possible.
- Certain characters have different character representations in different code pages. This might present problems when logging in with a different terminal than you normally use, for example, while traveling internationally. Avoid the use of such characters, when necessary.

If you share the RACF database with downlevel systems that do not support special characters in RACF passwords, or if you use a mix of applications that do and do not support special characters in passwords, do not activate the SETROPTS PASSWORD(SPECIALCHARS) option.

Assigning Password Phrases

You can assign a password phrase for a user using the PHRASE operand of the ADDUSER or ALTUSER command. This enables the user to authenticate using a password phrase instead of a password when using an application that supports password phrases.

```
ALTUSER ARUNDATI PHRASE('g0dofsm@llthings')
```

A password phrase is a character string consisting of mixed-case letters, numbers, and special characters including blanks. Password phrases have security advantages over passwords in that they are long

enough to withstand most hacking attempts, yet are unlikely to be written down because they are easy to remember.

You can define a user who has no password by omitting the PASSWORD operand on ADDUSER or by specifying NOPASSWORD on either ADDUSER or ALTUSER. This allows you the option of defining a user that can only authenticate with a password phrase, which is generally stronger than a password. The ability to define a "phrase-only" user will depend on whether the user uses any applications which check passwords, but do not support password phrases.

RACF enforces a basic set of syntax rules to increase the strength of password phrases. These syntax rules apply to all password phrases and you cannot alter or avoid them. However, you can add password phrase syntax rules to impose additional restrictions when your installation tailors the new-password-phrase exit (ICHPWX11).

When the new-password-phrase exit (ICHPWX11) is present and allows it, the password phrase can be 9-100 characters. When ICHPWX11 is not present, the password phrase must be 14-100 characters. Contact your system programmer to find out if your installation uses the new-password-phrase exit (ICHPWX11) or see *z/VM: RACF Security Server System Programmer's Guide* for programming details.

Password Phrase Syntax Rules:

- Maximum length: 100 characters
- Minimum length:
  - 9 characters, when ICHPWX11 is present and allows the new value
  - 14 characters, when ICHPWX11 is not present
- Must not contain the user ID (as sequential uppercase or sequential lowercase characters)
- Must contain at least 2 alphabetic characters (A-Z, a-z)
- Must contain at least 2 non-alphabetic characters (numerics, punctuation, or special characters)
- Must not contain more than 2 consecutive characters that are identical.
- Must be enclosed in single quotation marks, with single quotation marks within the password phrase doubled. Note that password phrases must be unquoted when RACF prompts at logon.
- Must not contain forward slashes, nulls (X'00'), or leading or trailing blanks

If the new-password-phrase exit (ICHPWX11) is present, it can reject the specified password phrase. Password phrases shorter than 14 characters will be rejected by RACF unless ICHPWX11 is present and allows the new value (minimum of 9).

If the specified password phrase is accepted, it is made the user's current password phrase and when SETROPTS PASSWORD(HISTORY) is in effect, it is added to the user's password phrase history.

See *z/VM: RACF Security Server Command Language Reference* for details about using the PHRASE operand of the ADDUSER and ALTUSER commands.

Setting the Maximum and Minimum Change Interval

If you have the SPECIAL attribute, you can specify the INTERVAL and MINCHANGE suboperands of the SETROPTS PASSWORD command. The INTERVAL suboperand specifies the system default for the maximum number of days that a user's password and password phrase remain valid. The MINCHANGE suboperand specifies the system default for the minimum number of days that must pass between a user's password (and password phrase) changes.

The following example specifies that each user's password and password phrase remain valid for 60 days (as long as the system default for these users remains 60 days) and that no user can change their password or password phrase more often than every 30 days (as long as the system default for these users remains 30 days).

```
SETROPTS PASSWORD(INTERVAL(60) MINCHANGE(30))
```

These values becomes effective immediately as:

- The default value for new users whom you define to RACF through the ADDUSER command
- The upper limit for users who specify the INTERVAL operand on the PASSWORD command

The initial system default is 30 days for the maximum change interval (INTERVAL) and 0 days for minimum change interval (MINCHANGE). The value MINCHANGE(0) allows users to change their passwords and password phrases more than once each day.

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).

Restrictions:

1. When you change the SETROPTS PASSWORD(INTERVAL) value, the password interval set in each user's profile is not changed. If a user's INTERVAL value in the user's profile (as set using the PASSWORD command) is different than the SETROPTS value, RACF expires the password or password phrase at the shorter interval of the two values.
2. Avoid setting the MINCHANGE value higher than any individual user's INTERVAL value (as set using the PASSWORD command). If you do, RACF expires the user's password or password phrase when the MINCHANGE period elapses, not when the user's INTERVAL elapses. Users cannot change their own passwords or password phrases until the MINCHANGE period elapses, even when the user's INTERVAL value defines a shorter period than the MINCHANGE value.

User consideration: Users who attempt to change their passwords or password phrases before the minimum change interval elapses are notified of their change failures but are *not* notified of the reason. The reason for the failure is withheld in the event of unethical user behavior, particularly by outside users or hackers who might exploit the information.

Extended Password Processing

The following options apply both to passwords and password phrases. These functions require SETROPTS INITSTATS to be in effect. See “Collecting LOGON Statistics” on page 77 for more information.

If you have the SPECIAL attribute, you can specify the WARNING/NOWARNING, HISTORY/NOHISTORY, and REVOKE/NOREVOKE options of the SETROPTS command.

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 accesses the system 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 (1-32) that RACF saves for each user and compares with an intended new value. When RACF finds a match with a previous value, or with the current password, RACF rejects the new intended value.

  Example: If you specify 12 for your HISTORY number, RACF saves up to 12 passwords for each user.

  ```
  SETROPTS PASSWORD(HISTORY(12))
  ```

  If you increase the HISTORY number, RACF saves and compares that number of passwords to the new intended value. If you subsequently reduce the HISTORY number, any previous passwords stored in the user profile in excess of the newly specified HISTORY number are not deleted and continue to be used for comparison. For example, if you specify 12 for your HISTORY number and subsequently reduce it to 8, RACF compares the old passwords 9-12 with the new intended value.

NOHISTORY specifies that new passwords are compared only to the current password. Any prior history information in the user profile 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. The following example specifies that RACF is to allow 4 consecutive invalid password attempts. A fifth invalid password attempt revokes the user ID:

```
SETROPTS PASSWORD(REVOKE(4))
```

If a user has both a password and a password phrase, the invalid attempts are cumulative. In this example, if the user entered 2 invalid passwords and then 3 invalid password phrases, he would be revoked.

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 invalid passwords are ignored.

User IDs which have neither a password nor a password phrase are not revoked based on consecutive incorrect passwords and password phrases.

Encryption of RACF User Passwords

RACF keeps user authentication data secure when it is stored in the RACF database. This authentication data can be a password or a password phrase. Passwords and password phrases are referred to collectively as "passwords".

RACF supports several different methods to encrypt authentication data:

- Masking
- The data encryption standard (DES) algorithm
- An installation-defined method that is implemented that uses the ICHDEX01 exit
- The KDFAES (key derivation function with AES256) algorithm (for passwords)

Encoding functions that are performed by RACF are:

- Data encoding
- 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 more details, see *z/VM: RACF Security Server System Programmer's Guide*.

Comparison means that, given authentication data as entered by a user (in clear text form) and given that data as stored in the RACF database in encoded form, an indication whether they are equal or not is returned.

By default, RACF uses the DES algorithm to encrypt and compare authentication data.

The DES, masking, and installation-defined methods are collectively referred to as "legacy" methods in some contexts. When KDFAES is not enabled, the presence and function of the ICHDEX01 exit determines which of these algorithms is active.

Guideline: Do not run your system with an ICHDEX01 exit unless you are using it to implement your own encryption algorithm.

For more information about the ICHDEX01 password authentication exit, see *z/VM: RACF Security Server System Programmer's Guide*.

The following SETROPTS command is used to enable KDFAES:

```
SETROPTS PASSWORD(ALGORITHM(KDFAES))
```

Note: Review the Planning Considerations for enabling KDFAES in *z/VM: RACF Security Server System Programmer's Guide* prior to enabling KDFAES.

KDFAES is the strongest algorithm and is recommended to be used when possible. This algorithm is resilient against offline brute-force password attacks if your RACF database or one of its copies is compromised.

When KDFAES is enabled, existing DES and installation-encoded passwords continue to evaluate correctly. When they are changed, they are encrypted using KDFAES. When KDFAES is active, the masking algorithm is no longer used to evaluate legacy passwords, and masked passwords must be changed by the administrator after KDFAES is enabled.

KDFAES passwords require more space in the RACF database than the legacy methods require. This changed format requires updates from some other software applications for them to keep functioning under the new algorithm. Be sure that you have all the available updates before you enable KDFAES.

If you need to disable KDFAES, the following SETROPTS command can also be used:

```
SETROPTS PASSWORD(NOALGORITHM)
```

After KDFAES is disabled, all legacy rules apply, however, KDFAES passwords continue to evaluate correctly. When they are changed, they are encrypted using the legacy algorithm in effect, which is determined by the presence and function of ICHDEX01.

Performing a Password Conversion

After KDFAES is enabled, you might want to strengthen passwords immediately, rather than wait for users to change them on their normal schedule. For example, you might want to prove compliance with a security policy that requires the strongest possible algorithm. The PWCONVERT operand of the ALTUSER command can be used. PWCONVERT converts a DES password and DES password history entries to KDFAES format without requiring the password to be changed.

A sample SQL/DS query against IRRDBU00 output is provided to report users that have a legacy format current password or phrase. This is in file RACDBUQR SAMPLE. See "Using the Database Unload Utility Output with SQL/DS" on page 275 for examples of using the IRRDBU00 output with SQL/DS.

For more information about the ALTUSER PWCONVERT function see *z/VM: RACF Security Server Command Language Reference*.

Notes:

1. Conversion assumes that the existing password and password history are in DES format. Do not perform the conversion if you have passwords that are masked or installation-encoded.
2. Conversion does not affect password phrases or phrase history.
3. If an older copy of your RACF database containing DES passwords is compromised and an attacker is able to guess a password value, a conversion to KDFAES will not protect against the attacker using that password to gain access to your system. If you suspect your database has been compromised, user passwords should be changed as soon as possible. The EXPIRED operand of the ALTUSER command can be used to force a user to change their password when the user next logs on.

Multi-Factor Authorization

On z/VM, Multi-Factor Authentication (MFA) is provided in an out-of-band (OOB) way. The following is the process for a user to authenticate with MFA:

1. The user must authenticate with the MFA server and get an MFA credential. This credential is obtained from a secure web page after entering the required factors for the selected policy. The available policies and their factors are configured and managed on the IBM MFA server. See IBM Z Multi-Factor Authentication (www.ibm.com/docs/en/zma/2.3.0).

2. The user must authenticate with z/VM by providing their user ID and the MFA credential. The characteristics of the MFA credential can be configured at the IBM MFA server (see Note “2” on page 75).
3. RACF then allows or disallows the authentication based on the decision of the MFA Server.

Note:

1. A user ID that is enabled for MFA and is enabled for PWFALLBACK must explicitly use the LOGON FALLBACK option if a FALLBACK logon is desired.
   - Their default logon method is the MFA process described above. If the MFA server fails the request, the logon fails and the provided credential is not re-evaluated as a password, password phrase or pass ticket.
   - The user must by-pass MFA by adding the FALLBACK option to their CP LOGON command and use their password, password phrase or pass ticket as the credential to logon. If the user is not allowed to use the FALLBACK option, RACF will disallow the logon without evaluating the credential. If the user is allowed to use more than one of these methods to logon, RACF will evaluate the passed-in credential using the same methods/order as it does today. The created SMF 80 JOBINIT record contains the authentication method that was used to authenticate this user.
2. The characteristics of the token can be configured at the MFA server to mirror the different types of currently accepted credentials, such as upper case or mixed case passwords or passphrases.

Considerations for the RACF Password and Password Phrase

The concept of a protected user has been extended to incorporate MFA. A protected user does not have a password, a passphrase, and is not enabled for MFA. Note that any user is by default a protected user unless a password or password phrase is explicitly assigned, or the user is enabled for MFA. That is, when a user profile is created, there is no default value assigned to the password or password phrase, and the user is disabled for MFA. The user is a protected user and cannot log on directly, but can log on using the XAUTOLOG and LOGONBY commands.

When the user is assigned the NOPWFALLBACK attribute, the password/phrase cannot be used to log on. The password should be removed when it is not necessary for system maintenance, recovery or MFA application bypass.

When the user is assigned the PWFALLBACK option, the user must maintain the password. The user's password can be changed using the PASSWORD or ALTUSER command or when performing a FALLBACK logon; it can not be changed during an MFA logon. An expired password does not prevent MFA logons and can only be changed on the next FALLBACK logon.

MFA Application Bypass

In some cases it may be desireable to bypass MFA and use the password or password phrase in selected z/VM applications. The RACROUTE macro includes a PASSCK=NOMFA option to allow applications to authenticate users against their password or password phrase regardless of the user's MFA enablement status. This function is equivalent to the user's FALLBACK option at LOGON in that MFA is not used to validate the password or password phrase.

Preparing to Use Multi-Factor Authorization

Complete the following steps to use IBM Multi-Factor Authentication for z/VM with RACF:

1. Install and configure the IBM MFA product. See IBM Z Multi-Factor Authentication (www.ibm.com/docs/en/zma/2.3.0).
2. Set up the RSCS DNS server, if necessary, and if domain names will be used for server definitions. See Setting Up the RSCS Domain Name Server in *z/VM: RSCS Networking Planning and Configuration*.
3. Configure the MFA CONTROL file. For more information, see “MFA CONTROL File” on page 76.

4. Use the ALTUSER MFA command to configure one test user to be enabled for MFA authentication in order to test and verify MFA behavior. For more information, see ALTUSER (Alter User Profile) in *z/VM: RACF Security Server Command Language Reference*.
5. Configure users to have the MFA PWFALLBACK value for system maintenance and recovery purposes, as appropriate.
6. Configure other selected users to be enabled for MFA authentication.

MFA Policies and Valid Factors

RACF for z/VM does not control or manage the policies and valid factors for MFA enabled users. These settings are controlled by the IBM MFA server and are configured on the MFA server. See IBM Z Multi-Factor Authentication (www.ibm.com/docs/en/zma/2.3.0).

MFA CONTROL File

During initialization, RACF uses the MFA CONTROL file to determine to which MFA server(s) it should send MFA requests for user authentication. The file must reside on RACFVM's A-disk. RACF processes the SERVER configuration records in sequence until it finds an MFA server that responds in time. If the currently selected MFA server is no longer responding, processing of the MFA CONTROL file is repeated from the beginning.

The following example is an MFA definition contained in an MFA CONTROL file:

```
*
MODE OOB
SERVER mfa-server.mfa-domain.org 6787 SAMPLE1 SIMPLEsecret TCPIP 900
SERVER 10.1.2.3   6787 SAMPLE1 SIMPLEsecret TCPIP 900
SERVER 127.0.0.1  6787 SAMPLE1 SIMPLEsecret
```

Three types of records are allowed in the MFA CONTROL file:

Comment records
begin with an asterisk. Empty lines are ignored.

MODE records
define the mode of operation. MODE records accept only the MODE OOB (out-of-band) parameter. The MODE OOB parameter causes out-of-band authentication to be accepted. The user authenticates to the MFA server and uses the provided MFA credentials to log on to z/VM.

SERVER records
Each SERVER record describes a single MFA server instance. Multiple MFA servers can be defined via multiple SERVER records. SERVER records accept the following parameters:

- MFA server host name (or IP address)
- MFA server port number (1-65535)
- ESM identifier (maximum of eight characters)
- ESM secret (maximum of 32 characters)
- TCP/IP user ID (optional, defaults to TCPIP)
- port number of RSCSDNS (optional, defaults to 900)

Usage Notes for the MFA CONTROL File

1. If a host name is specified for the target MFA server, the correct configuration of the RSCSDNS component is required. If the MFA server IP address is specified, no DNS lookups via RSCSDNS are performed. The defined host name or IP address must match the host name in the TLS certificate.
2. The ESM identifier and the ESM secret must match the values defined on the MFA server.
3. The optional TCP/IP user ID is used for all network communication, DNS lookups via RSCSDNS, and communication with the MFA server. If it is not specified, the default TCP/IP user ID of "TCPIP" is used.

4. If a host name is specified for the MFA server, the DNS lookup is performed via the RSCSDNS server which listens on the port defined in the MFA CONTROL file. If you specify the RSCSDNS port number, the TCP/IP user ID must also be specified. The default port number used is 900.
5. The MFA CONTROL file is re-read after the connection to the current MFA server is lost; therefore, editing an MFA CONTROL file during RACF operations will not lead to an immediate change. Consider stopping and starting the RACF server if you want to activate the new MFA CONTROL file immediately.

Revoking Unused User IDs

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)
```

Note:

1. New users who never use the system are not revoked because of inactivity.
2. If a user has not logged on (or submitted a job) in 31 days, and you issue the SETROPTS INACTIVE(30) command, that user will be considered revoked. However, the user will not actually be revoked and the output of the LISTUSER command will 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 once again start using the system (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 option cannot be used. See “Collecting LOGON Statistics” on page 77 for more information.

Collecting LOGON Statistics

When work enters the system, such as when a user logs on or a batch job is submitted, RACF creates a security environment for the appropriate user ID. The process of creating this security environment is referred to as "RACINIT", after the RACF macro instruction that is issued within a RACF service machine to create the security environment.

A user with the SPECIAL attribute can instruct RACF to record statistics during RACINIT processing. The statistics RACF maintains are:

- The date and time RACINIT is issued for a particular user
- The number of RACINITs for a user to a particular group
- The date and time of the last RACINIT for a user to a particular group.

The statistics must be maintained if you intend to use the INACTIVE, REVOKE, and WARNING options.

If, for your system, you do not need all the statistics, you do not need to use the above four options, and you have the SPECIAL attribute, you may issue SETROPTS NOINITSTATS which will reduce the RACF database I/O associated with RACINIT function.

If NOINITSTATS is in effect, the INACTIVE, REVOKE, 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 RACINIT statistics as shown in the following example:

```
SETROPTS INITSTATS
```

INITSTATS is in effect when a RACF database is first initialized using IRRMIN00.

Limiting When a User Can Access the System

Installations can restrict 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 restrict a user from entering the system, use the WHEN operand on the ADDUSER and ALTUSER commands. For example:

```
ADDUSER USER12 WHEN(DAYS(WEEKDAYS) TIME(0700:1700))
```

specifies that USER12 can enter the system only on weekdays between the hours of 7:00 a.m. and 5:00 p.m.

Similarly, in order 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, for a profile in the TERMINAL class:

```
RDEFINE TERMINAL TRM07C WHEN(DAYS(WEEKDAYS))
```

specifies that terminal TRM07C can be used at any time during the week, but not at all during the weekend. (When specifying the RDEFINE command, TIME(ANYTIME) is the default.)

Note: Protection for the TERMINAL class may be unreliable. IP addresses are unpredictable and modern local non-SNA terminals are provided by OSA-ICC, which simply accepts TN3270 sessions and converts them to local terminals, as do IBM 2074 Controllers. There do exist traditional 3174-like console controllers but they also have IP capability.

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 instruction) 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/VM: RACF Security Server Command Language Reference*.

User or Terminal Time and/or Day-of-Week Checking

The time and day-of-week checking for users and terminals applies when users log on to terminals from z/VM. User verification processing includes the following time and 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 the user is logging on from (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 the user is logging on from (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 Shared User IDs

Using the RACF LOGON BY function, you can define shared user IDs. Multiple users can log on to another user ID using their own password. This function uses:

1. The BY option of the LOGON command to specify the surrogate user
2. The SURROGAT class to perform authorization checks for logons to shared user IDs

RACF allows one user ID to log on to a shared user ID if that user has at least READ access to the SURROGAT profile named LOGONBY.*shared_userid*.

The SURROGAT profile also controls the ability to issue the CP FOR command against the same target user ID. See “Protecting the FOR Command” on page 145 for more information. The SURROGAT profile is used when LOGONBY authority is checked by DIAGNOSE X'88'. See “Protecting the DIAGNOSE X'88' Subcodes” on page 147 for more information.

To understand the function, you need to become familiar with the following terms:

shared user ID
User ID that has the capability of being logged onto by a different user.

surrogate user
Person logging on to the shared user ID.

direct logon
A “traditional” logon, in which you log on to your own user ID.

shared logon
A logon in which a surrogate user uses the BY option of the LOGON command to logon to a different user ID. The surrogate user operates with the RACF authority of the shared user ID.

With this support, you can share user IDs without compromising system security. To protect system security:

- The surrogate user's password is used to access the shared user ID. This avoids the need for users to share passwords.
- An audit trail identifies the surrogate user whenever work is performed by the shared user ID.

Interaction between RACF and z/VM

VM provides the LOGONBY directory statement to authorize shared LOGONs when no external security manager is installed. A description of how this statement is used when RACF is installed follows. For details on z/VM's LOGONBY directory statement, see *z/VM CP Planning and Administration*.

When RACF Is Active

When RACF is active, VM ignores the LOGONBY directory statement and leaves the access decision to RACF. RACF determines authorization based on the SURROGAT class profiles.

When RACF Is Inactive

When RACF has been set inactive by the SETRACF INACTIVE command, it defers the shared LOGON authorization decision to z/VM.

- If the surrogate user ID is specified on the shared user ID's LOGONBY directory statement, z/VM checks the password supplied by the surrogate user against the password in the surrogate user's directory entry. If these checks succeed, the shared LOGON is allowed. Since the CP directory does not support password phrases, a password must be used to LOGON to a shared user ID when RACF is inactive.

- If the user is authorized in the z/VM directory but not in the appropriate RACF SURROGAT profile, the shared logon succeeds when RACF is inactive. Once RACF is active again, it audits the user as a normal shared user ID for the duration of the logon.

When the RACF Service Machine Is Uninitialized or Unresponsive

When the RACF service machine is uninitialized or unresponsive, no users are allowed to log on to the system, except for the following:

- Any RACF service machine
- The primary system operator or any of the defined alternate system operators, if there is no system operator currently logged on.

For these user IDs, the same considerations apply as when RACF is inactive. See "When RACF Is Inactive" on page 79.

Note: By coding LOGONBY directory statements for the system operator user IDs or for RACF service machines, you still allow the use of LOGON BY for these user IDs when the RACF service machine is experiencing problems. This could be useful for recovery purposes.

Setting Up the LOGON BY Function

To let users share user IDs, perform the following tasks:

1. Define profiles of the form LOGONBY.*shared_userid* in the SURROGAT class for each user ID that is to be shared.
2. Permit specific users to the appropriate SURROGAT profiles.
3. Activate the SURROGAT class.

RACF checks for a SURROGAT profile on *all* logon attempts, both shared and direct, To improve performance, you may want to RACLIST the SURROGAT class. To do this, enter:

```
SETROPTS RACLIST(SURROGAT)
```

If the SURROGAT class is not active or the LOGONBY.*shared_userid* profile is not defined, the shared logon fails with:

```
RPIMGR065A Surrogate user authorization failed for user ID userid
```

Note: SURROGAT checking also fails under these circumstances in the context of DIAGNOSE X'88' and the FOR command.

Allowing Access to a Shared User ID

You can let a user ID log on to a shared user ID by defining the SURROGAT profile and permitting that user ID to that SURROGAT profile. For example, to allow BRUCEW to log on to the shared user ID DIRMAINT, the administrator can permit BRUCEW to the correct SURROGAT profile as follows:

```
RDEFINE SURROGAT LOGONBY.DIRMAINT UACC(NONE)
PERMIT LOGONBY.DIRMAINT CLASS(SURROGAT) ID(BRUCEW) ACCESS(READ)
```

Direct and Shared Logon

You can allow users to log on to a shared user ID directly or as shared.

Logging On as Shared

By default, a user ID that is defined as shared may not be logged on to directly.

For example, a service machine such as DIRMAINT needs an audit trail to identify which user has logged on to it. Therefore, users should not be allowed to log on directly.

Note: Define shared service virtual machines as NOPASSWORD and NOPHRASE user IDs. This provides another layer of protection against the user ID being logged onto directly, and also prevents the user ID from being revoked due to excessive incorrect password and password phrase attempts. See the NOPASSWORD operand of the ADDUSER and ALTUSER commands in *z/VM: RACF Security Server Command Language Reference*.

However, because the administrator permitted BRUCEW to the LOGONBY.DIRMAINT profile in the SURROGAT class, BRUCEW can log on as shared to the user ID DIRMAINT with the BY option of the LOGON command.

Logging On Directly

You have the option of letting users log on to a shared user ID directly. This may be desirable in some cases.

For example, manager DONNA wants her secretary BRADPITT to log on to her user ID, but also needs to log on to her own user ID. You can authorize this by defining and permitting the manager to the SURROGAT profile as follows:

```
RDEFINE SURROGAT LOGONBY.DONNA UACC(NONE)
PERMIT LOGONBY.DONNA CLASS(SURROGAT) ID(DONNA)    ACCESS(READ)
PERMIT LOGONBY.DONNA CLASS(SURROGAT) ID(BRADPITT) ACCESS(READ)
```

In this case, DONNA can log on to her own user ID whenever she needs to use it.

Simplifying SURROGAT Administration for System Maintenance IDs

By using RACF variable names in a SURROGAT profile name, you can define a single SURROGAT profile to control the ability for system programmers to LOGON BY to all of your maintenance IDs. To grant access to the SYSPROGS group, perform the following tasks:

1. ```
   SETROPTS GENERIC(SURROGAT)
   ```

   (must be done before defining the profile)

2. ```
   RDEFINE RACFVARS &MNTIDS ADDMEM(MAINT TCPMAINT RSCS PERFSVM)
   RDEFINE SURROGAT LOGONBY.&MNTIDS UACC(NONE)
   PERMIT LOGONBY.&MNTIDS CL(SURROGAT) ID(SYSPROGS) ACCESS(READ)
   SETROPTS CLASSACT(SURROGAT RACFVARS)
   ```

3. ```
   SETROPTS RACLIST(SURROGAT RACFVARS)
   ```

   or, if already RACLISTed:

   ```
   SETROPTS RACLIST(SURROGAT RACFVARS) REFRESH
   ```

To add another maintenance ID to the list:

```
RALTER RACFVARS &MNTIDS ADDMEM(TCPIP)
SETROPTS RACLIST(RACFVARS SURROGAT) REFRESH
```

Note:

1. Issue SETROPTS RACLIST REFRESH any time you update the RACFVARS or SURROGAT classes (because they were both RACLISTed).
2. Note that MAINT, TCPMAINT, RSCS, and PERFSVM are NOT in the group so you will not be allowed to logon directly to the ID, and MUST use LOGON BY.

Special Considerations

You need to consider the following when using the LOGON BY function.
```

Ownership Considerations

If the shared user ID belongs to a person who wants to let other people log on to that ID, you can make that person's user ID the owner of the SURROGAT profile. The owner can permit people to log on to the user ID without requiring intervention from an administrator.

For example, you can make DONNA the owner of her SURROGAT profile as follows:

```
RALTER SURROGAT LOGONBY.DONNA OWNER(DONNA)
```

You should be aware that when a surrogate user logs on to that user ID, the surrogate user has the authority to permit other users to log on to the shared user ID.

Authorization Considerations

If you change the surrogate user's authorization to a shared user ID's SURROGAT profile while that shared user is logged on, the change does not take effect during the current logon session. It takes effect the next time the shared user ID logs on.

For example, while BRUCEW is logged on to DIRMAINT, the security administrator changes his access from READ to NONE in the SURROGAT profile. BRUCEW can use DIRMAINT for the remainder of the session, but cannot log on to the ID again.

Terminal Considerations

If the TERMINAL class is active when a surrogate user tries to log on to a shared user ID, both the shared user ID and the surrogate user must have access to the terminal being used.

Security Label Considerations

If the SECLABEL class is active, both the shared user and the surrogate user must be permitted to the appropriate SECLABEL profile. RACF uses the SECLABEL from the first place it is found subject to the following order:

1. The SECLABEL specified on the LOGON command
2. The SECLABEL specified as the surrogate user's default SECLABEL in its USER profile
3. The SECLABEL specified as the shared user ID's default SECLABEL in its USER profile

If no SECLABEL is found in any of these places, the SETROPTS MLACTIVE setting determines whether the user can logon without a SECLABEL. If MLACTIVE(FAIL) is in effect, a user cannot log on without a SECLABEL.

If the SECLABEL class is active and a SECLABEL exists in the SURROGAT profile, but a SECLABEL was neither specified on the command line nor in the surrogate user's USER profile, the shared logon attempt fails. In this case, the user logging on receives message RPIMGR065A and the security console receives message ICH408I indicating that the submitter is not authorized by the user.

General Considerations for User ID Delegation

- 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 to resource profiles rather than users.
- 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.

- Once 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 while JOIN does not.

Figure 6 on page 83 shows delegating authority in another way.

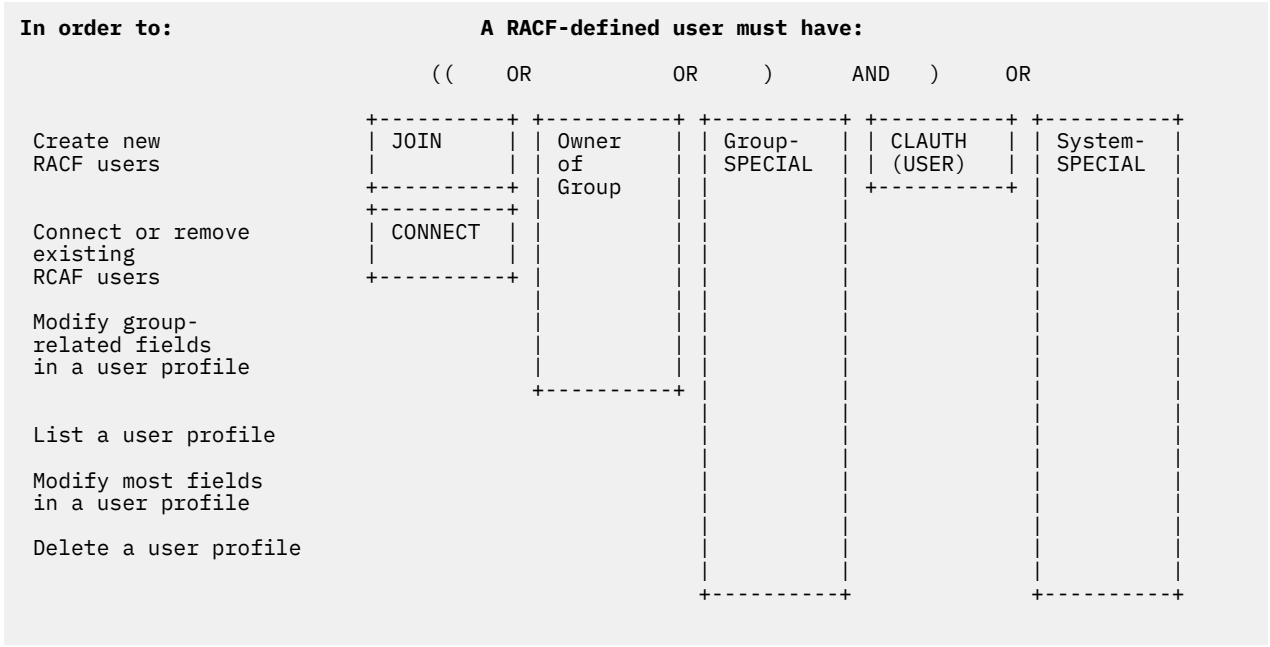


Figure 6. Delegating Authority (User Profiles)

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 system-SPECIAL.

Summary of Steps for Defining Users on z/VM

This summary presents the steps required by RACF and related IBM licensed programs to define users to RACF on z/VM. 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. See “Summary of Steps for Defining a RACF Group” on page 95.
   - Decide which user ID to assign to the user.

     Note:

     a. This must match the user's directory entry.

     b. If your installation is using access control groups, (ACIGROUP control statement in directory entries), see “Use of ACIGROUP Control Statements” on page 94.
   - Decide which user or group is to be the owner of the user profile. (If the owner is a user, provide him or her with the necessary information for managing the new profile.)
   - Decide on the authentication method for each user: password, password phrase, MFA, or any combination of the three.
     - If the user will have a password, decide on the initial password that will be assigned to the user. You should specify a non-trivial password.
     - Decide if the user should be required to use a password phrase to access the system and, if so, choose the user's initial password phrase. You can define a user that can only authenticate by using a password phrase, which is generally stronger than a password, by specifying NOPASSWORD and assigning a password phrase. The ability to define a "passphrase-only" user depends on whether the user uses any applications which check passwords, and which do not support password phrases.
     - If the user is going to use MFA, decide whether PWFALLBACK should be enabled for the user. With MFA, an initial password or, preferably, passphrase should be set and changed as soon as possible by the user. Also, the user must be enrolled in the correct policy in the IBM Multi-Factor Authentication server before any successful authentication can be performed
   - Determine if the user's access to the system should be restricted 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 default security label to assign to the user.
2. Create the user profile.

   Note: This can be done using any of the following methods:
   - Issuing the ADDUSER command.
   - If DirMaint is installed, using the dual registration panels supplied with RACF for z/VM.
3. Create minidisk profiles for each of the user's minidisks.

   Note: This can be done using one of the following methods:
   - Using the RDEFINE command:

     ```
     RDEFINE VMMDISK LWARD.191 UACC(NONE)
     ```
   - Using the dual registration panels.

   Note: Discrete profiles are recommended for the VMMDISK class. For more information, see “Protecting z/VM Minidisks” on page 152.
4. If the user will be enrolled in an SFS file pool, create the appropriate generic profiles. For example, if ANDREW will be enrolled in file pool POOL1, enter:

```
ADDDIR POOL1:ANDREW.** OWNER(ANDREW)
ADDFILE * * POOL1:ANDREW.** OWNER(ANDREW)
```

5. If users at your installation manage their own resource profiles, provide the user with the necessary information for doing so. For example, the user might need to use portions of the *z/VM: RACF Security Server General User's Guide*.
6. 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 provide the user with information for working with those profiles. For example:

```
TAPEVOL class
TERMINAL class
VMSEGMT class
```

7. 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.

     Note: This can be done using the CONNECT command.
   - 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.

     Note: This can be done using the PERMIT command.

Deleting a User

Summary of Steps for Deleting Users on z/VM

This summary presents the steps required by RACF and related IBM program products to delete users from RACF on z/VM. 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. Check one of the following two sections to delete the user profile and all occurrences of the user ID from the RACF database:
   - "Running the RACFDEL and RPIDELU EXECs" on page 85
   - "Deleting a User Manually" on page 86.

Running the RACFDEL and RPIDELU EXECs

RACFDEL and RPIDELU are sample EXECs that are shipped on the RACF product tape.

RACFDEL uses the work file produced by the RACUT100 EXEC (IRRUT100 SYSUT1) as input, and produces RACF commands in a file called RDEL CMDS as output.

Requirements for using RACFDEL are:

- You must have the system-SPECIAL attribute.
- RACFDEL cannot be run from either the primary or the backup RACF service machine.
- Make sure that you have READ access to the file containing current output from the RACUT100 EXEC (the file is IRRUT100 SYSUT1).

RACFDEL prompts for:

- The user ID to be deleted.

- The user ID or group ID of a new owner for resources owned by the user being deleted. You can supply a single user ID to be the new owner of all the old user's resources, or you can supply a different user ID for each resource that needs a new owner.

RPIDELU executes the commands that RACFDEL produces. Like RACFDEL, you must have the system-SPECIAL attribute to use RPIDELU. Before you execute RPIDELU, *examine the RACF commands in RDEL CMDS carefully*. Some of the things you should look for are:

- REMOVE from default group

  Because RACFDEL cannot tell which group is the user's default group, RDEL CMDS will contain a REMOVE command to remove the user from his or her default group. When this command is executed, RACF will produce an error message. You can avoid this message (which can be ignored) by deleting the command that attempts to remove the user from his or her default group.
- RDELETE for SURROGAT, VMBATCH, and VMRDR profiles

  RDEL CMDS contains RDELETE commands to delete the profiles for the user's minidisks and SFS files and directories.

  RDEL CMDS does *not* contain commands to delete LOGONBY.*userid* (SURROGAT), batch (VMBATCH), or reader (VMRDR) profiles. If you want to delete profiles for these resources from the RACF database, add the appropriate RDELETE commands to RDEL CMDS.

See step 5 under "Deleting a User Manually" on page 86 for information about deleting BFS files and EXEC.U*uid* and EXEC.G*gid* profiles.

Deleting a User Manually

You can delete a user's user profile manually and remove all occurrences of the user ID from the RACF database.

1. If the user owns any resource profiles (the user's user ID appears in a qualifier of the profile name), delete the resource, or, for minidisks, transfer the minidisk to another user.

   Note:

   a. You can use the dual registration panels to transfer a minidisk to another user.

   b. You can use the following SEARCH command to identify which general resource profiles includes the user ID in the profile name.

   ```
   SEARCH  CLASS(class-name)  FILTER(**.userid.**)
   ```

   If you use the RAC command processor, the output of the SEARCH command is written to a file named RACF DATA. You can then edit this file to generate RDELETE commands to delete profiles. You should consider using this SEARCH command in any class with profile names that can include user IDs: FILE, DIRECTRY, SURROGAT, VMBATCH, VMCMD, VMMDISK, VMLAN, and VMRDR.

2. Work with the z/VM administrator to delete the user's entry from the z/VM directory.

   Note: If you use the dual registration panels to delete the user, the user's z/VM directory entry is also deleted.
3. Remove the user from any access lists in which the user's user ID is specified.

   Note:

   a. To research this step, use the IRRUT100 utility to list the occurrences of the user ID in the RACF database.

   b. To do this, use the DELETE operand on the PERMIT command.
4. If the user owns any RACF profiles, change the OWNER field of the profile.

Note:

a. To research this step, use the IRRUT100 utility to list the occurrences of the user ID in the RACF database.

b. To do this, use the appropriate RACF alter command, such as ALTUSER or RALTER.

5. If the user ID has a UID assigned to it in the OVM segment of its USER profile, check for any files in the BFS that are owned by the UID. These files will either need to be deleted or transferred to another UID using the chown command. For more information about chown, see *OpenExtensions for z/VM: Command Reference.*

   If the UID owns any set-UID or set-GID executable files, there may be profiles in the VMPOSIX class that need to be deleted or renamed appropriately. These profiles have the format:

   ```
   EXEC.U<uid>
   EXEC.G<gid>
   ```

   See "Protecting Set-UID and Set-GID Executable Files" on page 205 for details on these profiles.

6. After all occurrences of the user ID are deleted from the RACF database, delete the user profile.

   Note: This can be done using any of the following methods:

   - Issuing the DELUSER command
   - Using the dual registration panels.

Chapter 5. Defining Groups

This chapter provides in-depth information on defining groups for z/VM systems.

Note: See Chapter 12, “Controlling OpenExtensions and BFS Security,” on page 195 for information about RACF security at the Portable Operating System Interface (POSIX) 1003.1 level.

The group structure of RACF can be mapped into the organizational structure that exists at your installation. That is, RACF conforms naturally to a tree structure of groups, with each group except the highest (the IBM-supplied SYS1 group) having a superior, or owning, group. Groups can correspond directly to business entities such as divisions, departments, and projects; users can be connected to one or more groups.

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

When setting up RACF groups, you might want to keep in mind that the maximum number of users that you can connect to any one group is approximately 5900. See *z/VM: RACF Security Server System Programmer's Guide* for information about how to determine the exact maximum number.

Administrative Group: 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 could create this kind of group for each group administrator. Operating from such groups, the group administrators could then define other groups needed by their local users.

Holding Group: A popular technique that retains user definition centrally, yet allows 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 is of no real significance to the user.

Group administrators, to whom you then give CONNECT (but not JOIN) authority, 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/VM: RACF Security Server System Programmer's Guide*.

Functional Group: 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 will have to 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 Group: 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.195 is a VM minidisk, two groups could be defined, PAYREAD and PAYUPDTE, both of which would appear in the PAY.195 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 can consist of the following parts, or *segments*: a RACF segment, and optionally, an OVM segment.

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. You can also list the contents of an entire group profile, or the contents of individual segments of the group profile using the LISTGRP command. For information on how to use these commands, see *z/VM: RACF Security Server Command Language Reference*.

The RACF Segment in Group Profiles

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

group-name
Name of the group

OWNER
Owner of the group's profile

SUPGROUP
Name of the superior group

TERMUACC/NOTERMUACC
Indicates whether to allow access based on the UACC of the terminal profile (for terminals protected by RACF)

DATA
Installation-defined data

The OVM Segment in Group Profiles

You can use the OVM segment of the group profile to specify information about the RACF group members' POSIX group ID. Specifically, when you define a new OpenExtensions group or change OVM attributes for an existing group, you can specify the following information in the group's profile:

GID
Specifies the group's OpenExtensions group identifier

When a user is connected to a group that has a GID value assigned to it, that GID can be used in determining a user's access to files in the byte file system (BFS), in accordance with the rules of fileaccess defined by the POSIX standard. See “Defining OpenExtensions Groups” on page 201 for more information.

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 by way of 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 by way of field-level access control. For more information, see “Field-Level Access Checking” on page 210.

Group Naming Conventions

The group naming conventions are relatively simple:

- A group name must be from 1 to 8 characters long, 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.

Since there is a potential that two or more users might want to use the same group name (for example, ADMIN), you should adopt naming standards locally to prevent this. Consider, for example, 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 consisting 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.

With OpenExtensions,the group identifier (GID) is an integer value that defines a group. Although the same value can be used to define multiple groups, it is not recommended. If you use the same value, control at an individual group level is lost because the GID is used in OpenExtensions security checks. Groups with the same GID value would be treated as a single group during OpenExtensions security checks. See “Defining OpenExtensions Groups” on page 201 for more information.

Benefits of Using RACF Groups

This section describes some of the ways that RACF groups can be particularly useful.

Reducing the Effort of Maintaining Access Lists

Instead of adding and deleting users to the access lists of several profiles, consider creating a RACF group and placing it on the access list instead of the user IDs. Then, 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 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 system-SPECIAL to refresh the in-storage profiles.

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 profile(s) 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 system- or group-OPERATIONS attribute. Depending on the class of the resource, having 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, issue the following command:

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

To cause the user's access to start on a certain date, issue the following command:

```
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 describe the various aspects of group ownership, group authorities, suggestions for assigning group authorities, and the group terminal option.

Ownership of a RACF Group

Each group 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 creating a new group profile, or with the ALTGROUP command when altering an existing group profile. If you are the owner of a group (or if you are a connected user having 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 give that user the ability 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 39 on page 221.

Group Ownership of Profiles

You can assign a *RACF group* as the owner of a user profile, group profile, data set profile, minidisk profile, or any other 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 will have the authority of SPECIAL for all profiles owned by the group (see “User Attributes” on page 57) and will have 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 a user or by 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 allows 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 various group authorities are described in Table 11 on page 93.

Table 11. 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 get access to resources (such as minidisks, terminals, and SFS files and directories) the group is authorized to.	LDIRECT LFILE LISTDSD RLIST
CONNECT	A user with CONNECT group authority can connect users (who are already defined to RACF) to the group and assign USE, or CONNECT group authority to users in the group. CONNECT group authority includes the privilege of USE group authority.	All of the above, plus: • ALTUSER (only GROUP, AUTHORITY, or UACC operands) • CONNECT (all operands except SPECIAL, NOSPECIAL, OPERATIONS, NOOPERATIONS, AUDITOR, and NOAUDITOR) • LISTGRP (only group-name operand) • 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 USE and CONNECT authorities.	All of the above, plus: • ADDGROUP (all operands) • ADDUSER (all operands except OPERATIONS, SPECIAL, AUDITOR, and ROAUDIT) • ALTGROUP (SUPGROUP operand only; to change the superior group of a group, a user must have JOIN authority in one group and be owner of or be connected with the group-SPECIAL attribute to another group) • DELGROUP (all operands) • LISTGRP (only group-name operand)

For a list of the RACF commands that the group authorities allow users to issue, see Table 37 on page 220.

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 Delegation:** You can have one delegate (group owner) be responsible for the administration of a group, the users in the group, and the group resource profiles. The group owner, in this scheme, connects to the group with JOIN authority, defines the group resource profiles, and connects other users to the group with USE authority.
- **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 USE authority to all other users. In this way, the owner of the group can monitor the group, and the user with JOIN authority can monitor the users in the group.

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 VM 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 VM from terminals (either RACF-defined or otherwise) based on the universal access authority of the terminals.

List-of-Groups Authority Checking

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: On z/VM, a user's current connect group is the default group specified in his or her user profile.

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 id for Group B, but it does contain the group id 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. (See Chapter 5, "Defining Groups," on page 89 for more information on the scope of the group.)

For example, in Figure 5 on page 65 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.

We recommend 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.

Considerations for OpenExtensions z/VM

OpenExtensions z/VM groups are RACF groups that have a GID defined in the OVM segment of the group's profile. Authority checks for access to OpenExtensions VM byte file system files and directories use the GID in the user's current connect group and up to NGROUPS_MAX supplementary groups (if SETROPTS GRPLIST is active) to make group access decisions. Authority checks for other z/VM resources use the RACF current group and list-of-groups support. See the ICHNGMAX macro in *z/VM: RACF Security Server Macros and Interfaces* for more information on NGROUPS_MAX.

Use of ACIGROUP Control Statements

Attention:
The use of RACF groups is recommended rather than ACIGROUPs.

If a user's CP directory entry has an ACIGROUP control statement, the following considerations apply:

- If the SETROPTS GRPLIST option is in effect, the group specified on the ACIGROUP control statement for a user must be a RACF-defined group of which the user is a member.
- If the SETROPTS GRPLIST option is *not* in effect, the group specified on the ACIGROUP control statement for a user must be the user's default connect group.
- Profiles for the user's minidisks and virtual unit record devices are prefixed with the user's ACIGROUP name. For example, if user SUE's directory entry has an ACI group of GRP1, her 191 disk is protected by the following profile:

```
RDEFINE  VMMDISK  GRP1.SUE.191 UACC(NONE)
```

  and her virtual reader, printer, and punch are protected by:

```
RDEFINE  VMRDR  GRP1.SUE UACC(NONE)
```

- You can use the RACGROUP EXEC to determine a user's access control group:

```
RACGROUP userid
```

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.

     Note: 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.)
   - If your installation is using RACF to protect terminals, and the users in this group are terminal users who are to be restricted to specific terminals, consider specifying the NOTERMUACC operand on the ADDGROUP command.
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 STEVEH, LIZS, and GENEK to the DEPTA group and also give LIZS and STEVEH authority to add new users to the group, enter:

```
CONNECT (STEVEH LIZS) GROUP(DEPTA) OWNER(DEPTA) AUTHORITY(CONNECT)
CONNECT GENEK 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 GENEK's authority defaults to USE, GENEK can use any of the resources (for example, terminals) that belong to group DEPTA.
4. If the group requires access to RACF-protected resources, give the group the required access using the PERMIT command. For example:

```
PERMIT TERM007 CLASS(TERMINAL) ID(DEPTA) ACCESS(READ)
```

5. If the group requires access to OpenExtensions VM resources, alter the profile to include an OVM segment with a GID. For example:

```
ALTGROUP DEPTA OVM(GID(100))
```

Summary of Steps for Deleting Groups

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

1. Remove all users from the group.

   Note: You can use the REMOVE command to do this. Before removing a user from the user's default connect group, you must first connect the user to a new group (CONNECT command), then change the user's default connect group to the new group (ALTUSER command).
2. To research the following steps, use the IRRUT100 utility to list the occurrences of the group ID in the RACF database.
3. For each subgroup of the group to be deleted, change the subgroup's superior group to an existing group.

   ```
   ALTGROUP  subgroup-name  SUPGROUP(new-superior-group-name)
   ```
4. If the group is the owner of any profiles (the group's group ID 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 RACF alter command, such as ALTUSER, ALTGRP, or RALTER.
5. Remove the group from any access lists the group's group ID might be specified in.

   Note: To do this, use the DELETE operand on the PERMIT command.
6. If the group has a GID assigned to it in the OVM segment of its GROUP profile, check for any files in the BFS that are owned by the GID. These files will either need to be deleted or transferred to another GID using the `chgrp` or `chown` command. See *OpenExtensions for z/VM: Command Reference* for more information.

   If the GID owns any set-GID executable files, there may be profiles in the VMPOSIX class that need to be deleted or renamed appropriately. These profiles have the format:

   ```
   EXEC.Ggid
   ```

   See “Protecting Set-UID and Set-GID Executable Files” on page 205 for details on these profiles.
7. After all occurrences of the group ID are deleted from the RACF database, delete the group profile.

   Note: This can be done using the DELGROUP command.

Chapter 6. Defining Resources

Class-wide SETROPTS Options

Several RACF options of the SETROPTS command allow you to perform operations on groups of profiles on a class-wide basis. These options are listed in Table 12 on page 97. See *z/VM: RACF Security Server Command Language Reference* for additional information on these options. For a list of general resource classes supplied by IBM, see "Description of RACF Classes" in *z/VM: RACF Security Server Command Language Reference*.

Table 12. Class-wide SETROPTS options

Perform this task	**Using this SETROPTS operand**
Specify the names of the classes for which you want RACF to perform or cease auditing. See "Activating and Deactivating General Resource Classes" on page 99.	AUDIT operand
Specify those classes defined by entries in the class descriptor table for which RACF protection is or is not to be in effect. See "Activating and Deactivating General Resource Classes" on page 99.	CLASSACT operand
Activate or deactivate generic profile command processing for specified classes. See "Generic Profile Names" on page 102.	GENCMD operand
Activate or deactivate generic profile checking for specified classes. See "Generic Profile Names" on page 102.	GENERIC operand
Activate or deactivate sharing of in-storage generic profiles for the specified classes. See "SETROPTS GENLIST Processing" on page 113.	GENLIST operand
Specify classes eligible or ineligible for global access checking. See "Creating Global Access Checking Table Entries" on page 122.	GLOBAL operand
Audits access attempts to resources in specified classes according to the auditing-level specified. See "Allowing a Warning Period" on page 33.	LOGOPTIONS(ALWAYS/NEVER/SUCCESSES/ FAILURES/DEFAULT) operand
Activate or deactivate the sharing of in-storage profiles, both generic and discrete, for specified classes. See "SETROPTS RACLIST Processing" on page 114.	RACLIST operand
Record or not record statistical information for specified classes. See "Using SETROPTS STATISTICS for Statistics Collection" on page 115.	STATISTICS operand

Table 13 on page 98 lists the RACF commands you can use to work with general resource profiles.

Table 13. RACF Commands Used to Work with General Resource Profiles	
Activity	**Command**
Defining	RDEFINE
Changing	RALTER
Allowing or denying access	PERMIT with CLASS(*class-name*)
Searching	SEARCH with CLASS(*class-name*)
Listing	RLIST
Deleting	RDELETE
Managing on a class-wide basis	SETROPTS

Note: For the authority needed to issue any of these commands, see *z/VM: RACF Security Server Command Language Reference*.

For further information on working with RACF general resources, see Chapter 10, "Protecting z/VM Resources," on page 151.

Note: See Chapter 12, "Controlling OpenExtensions and BFS Security," on page 195 for information about RACF security at the Portable Operating System Interface (POSIX) 1003.1 level.

Defining Profiles for General Resources

To protect a general resource, use the RDEFINE command to define a general resource profile. You can also use the ISPF panels to define general resource profiles.

When you create a general resource profile, you must specify the class name and the profile name. For example:

```
RDEFINE  class-name  profile-name
```

Any time you wish to refer to the profile (for example, when changing its access list), you must give the profile name and class name.

Examples in this book also include the UACC (universal access authority):

```
RDEFINE  class-name  profile-name  UACC(universal-access-authority)
```

UACC is usually shown as NONE. This prevents all users not otherwise specified in the access list from accessing the resource.

Usually, you will also issue the PERMIT command to set up the access list in the profile. A sample PERMIT command is:

```
PERMIT  profile-name  CLASS(class-name)
        ID(user or group)  ACCESS(access-authority)
```

When you enter the RDEFINE command, you can specify much more than just profile name, class name, and UACC. In most cases, RACF provides appropriate defaults for this additional information. Where additional information is necessary for the profile (such as specifying the ADDMEM operand for resource grouping profiles), this book gives examples and describes appropriate values. Some of the additional operands that you might consider specifying are:

- OWNER—The user ID or group name of the owner of the profile
- NOTIFY—A user ID to be notified when access attempts fail
- AUDIT—Whether access attempts are to be logged, and if so, at which level.

Other operands are available. For a complete list of the operands for the RDEFINE command, see *z/VM: RACF Security Server Command Language Reference*.

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 named in the access list of that resource profile can still access the resource with the authority specified by UACC (unless the UACC is NONE). These users include users not defined to RACF.

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 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 14 on page 99 shows additional meanings for several access authorities for general resources.

Table 14. Access Authorities for General Resources

Authority	Meaning
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/or groups to access the resource. For generic profiles, only the profile owner, users with the system-SPECIAL attribute, and group-SPECIAL users whose groups own the profile have control over the resource profile and can authorize other users and/or 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 to list the profile.
CONTROL READ UPDATE	These access authorities allow listing of selected portions of the profile and grant resource access in a variety of ways, depending on the class.

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.

Activating and Deactivating General Resource Classes

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 VMMDISK and VMRDR resource classes.

```
SETROPTS CLASSACT(VMMDISK VMRDR)
```

It is not recommended that you activate all RACF classes. You should only activate 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. When using the SETROPTS CLASSACT(*) operand, RACF prevents you from accidentally activating classes that have a default return code of 8.

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. However, you can still define resource profiles to RACF using the ADDDIR, ADDFILE, and RDEFINE commands.

Special Considerations for OpenExtensions z/VM

The following classes are defined only for auditing OpenExtensions VM security events and are not used for authorization checking:

DIRACC
DIRSRCH
FSOBJ
FSSEC
PROCESS

No profiles can be defined in these classes. They are used to define the auditing options for OpenExtensions VM 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 class. For more information about the SETROPTS command, see *z/VM: RACF Security Server Command Language Reference*

Protecting General Resources with Discrete Profiles

A *discrete profile* protects a single resource. For example, if a resource requires special access authorization or unique logging information, you can protect it with a discrete profile.

You can protect a resource with a discrete profile by using the RDEFINE command with the resource's class name and profile name. For a description of the IBM-supplied general resource classes, see “Protecting General Resources” on page 15.

The following example shows how to protect SMITH's A-disk (which has a virtual address of 191) with a discrete profile:

```
RDEFINE  VMMDISK  SMITH.191  UACC(NONE)
```

Protecting General Resources with Generic Profiles

A *generic profile* protects one or more resources. Resources protected by generic profiles must have similar names and identical security requirements. For example, a generic minidisk profile can protect one or more minidisks.

If you use a generic profile on z/VM, follow the naming conventions for z/VM resources and the rules for defining generic profiles. To protect a resource with a generic profile, use the RDEFINE command and specify the resource's class name and generic profile name. The *generic profile name* must contain one or more generic characters (%, *, or &). For example, you can protect all z/VM minidisks owned by USERA with a generic profile as follows:

```
RDEFINE  VMMDISK  USERA.*  UACC(NONE)
```

Note: A SETROPTS GENERIC(classname) command must first be issued to turn generics on for the class, followed by a SETROPTS REFRESH command.

Protecting General Resources Using Models from Existing Profiles (or Access Lists in Existing Profiles)

You can protect a resource with a generic or discrete profile by using an existing generic or discrete profile as a model. Create the model by specifying the existing profile name on the FROM operand.

- As you create a new profile, you can specify FROM on the RDEFINE command to identify the profile to be copied. For example, to copy an existing generic minidisk profile called SMITH.* when creating a new minidisk profile named JONES.*:

```
RDEFINE  VMMDISK  JONES.*  FROM(SMITH.*) UACC(NONE)
```

  If the profile you are copying is not in the same class, specify the FCLASS operand. The following command copies information from profile TV1234 in class TAPEVOL:

```
RDEFINE  VMMDISK  JONES.*  FROM(TV1234)  FCLASS(TAPEVOL) UACC(NONE)
```

- When you work with an access list, you can specify FROM on the PERMIT command to identify a profile whose access list is to be copied. For example, to copy the access list of an existing minidisk profile called SMITH.* to an existing minidisk profile named JONES.*:

```
PERMIT  JONES.*  CLASS(VMMDISK)
        FROM(SMITH.*)
```

  If the profile you are copying is not in the same class, specify the FCLASS operand. The following command copies the access list from profile TV1234 in class TAPEVOL:

```
PERMIT  JONES.*  CLASS(VMMDISK)
        FROM(TV1234)  FCLASS(TAPEVOL)
```

Notes:

- The copied information is not necessarily identical to that specified in the FROM profile. See "Possible Changes to Copied Profiles When Modeling Occurs" on page 32.
- General information on generic profiles for general resources on z/VM is in "Generic Profile Names" on page 102.
- Information on global access checking is in Chapter 7, "Fast Authorization Using the Global Access Checking (GAC) Table," on page 121.

Access Authorization Checking for the General Resources

During access-authorization checking, RACF first checks for a discrete profile for a resource. 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. For a description of the search order RACF uses, see Table 16 on page 105.

Choosing Between Discrete and Generic Profiles in General Resource Classes

When you are creating profiles in the general resource classes you can create either a discrete or a generic profile.

Choose a *generic profile* to protect more than one resource with the same security requirements.

Note:

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 107.

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, Resource Group, and RACFVARS Profiles

Table 15 on page 102 gives some considerations for choosing among generic profiles, resource group profiles, and RACFVARS profiles.

Table 15. Choosing Among Generic, Resource Group, and RACFVARS Profiles

How to Choose	Reference
Use generic profiles when the names of the resources have logically matching characters.	"Generic Profile Names" on page 102 and *z/VM: RACF Security Server 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).	"Creating Resource Group Profiles" on page 110.
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 107.

Generic 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 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.

For generic profile naming of general resources, you can use an asterisk (*) within a profile name, representing one qualifier of a resource name. You can 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.

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, with examples, of how to specify * and **, see *z/VM: RACF Security Server Command Language Reference*.

Restricted Use of %* in General Resource Profile Names

The %* combination requires special attention.

New profiles with an ending %* are no longer allowed nor are profiles named %*. The RDEFINE command will return 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 will 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:

- The above considerations also apply to generic members of grouping classes.
- When creating the new profiles, consider using the FROM operand for continued use of the same access list.

For general resource classes

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 they will use in their invocations of the security product, they should be aware of the problems they will have in using *, %, or & in resource names. For more information, refer to *z/VM: Security Server 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 be able to 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 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 will never be 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 log on", not "allowed to read information".

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 RHONDA have the SPECIAL or group-SPECIAL attribute. If the GENERICOWNER option is in effect, and user RHONDA is the owner of a JESSPOOL profile called NODEA.RHONDA.**, JOE cannot create profile NODEA.RHONDA.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 * or **. More specific profiles are profiles that match the less specific *top* profile name character for character, up to the ending * or ** in the less specific name.

In a search for the less specific profile a match is found if *both:* of the following are true:

- The profile name ends in * or **
- All characters preceding the * or ** exactly match the corresponding characters in the resource name.

For example, to allow BOB to RDEFINE A.B in the JESSPOOL class, you need profile A.* in the JESSPOOL class, which is owned by BOB.

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 209.

Generic Profile Checking of General Resources

The rules for access-authorization checking of generic profiles for general resources are:

- Generic profiles are not checked unless generic profile checking is in effect for the class. To do this, issue the following command:

```
SETROPTS GENERIC(class-name)
```

- 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 Resources Protected by RACF Profiles” on page 309.
- If more than one profile covers a particular resource, RACF searches for profiles in the following order:
  1. Discrete profile
  2. Matching generic profiles (see Table 16 on page 105)

and stops at the first matching profile.

Table 16. Sample General Resource Profile Names (Most Specific to Least Specific)

Profile Name	Profile Type	Resources Being Accessed			
		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		
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, as follows:

```
SEARCH CLASS(VMMDISK) 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, take the following steps:

1. Find all profiles that match the resource name.

2. If no profile names match, check for profile names that include & (RACF variables). You must list the RACFVARS profile to determine the value of a RACF variable:

   ```
   RLIST  RACFVARS  variable-name
   ```

   Also, the SEARCH command will 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 111.

   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, then:
   - 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 with a generic character farther from the beginning of the name is used.

Note: If two profile names match except for one character position, the following is the order in which RACF searches for 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 searched:

```
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.

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 VMCMD class.

```
SETROPTS GENERIC(VMCMD)
```

If you specify GENERIC(*), you activate generic profile checking for all classes in the class descriptor table except resource group classes (such as GTERMINL).

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 VMCMD class.

```
SETROPTS NOGENERIC(VMCMD) GENCMD(VMCMD)
```

NOGENERIC and NOGENCMD are in effect when a RACF database is first initialized using IRRMIN00.

Note: Generic profiles are recommended only if you also specify SETROPTS GENLIST for the class.

Using RACF Variables in Profile Names (RACFVARS Class)

You can create profiles in the RACFVARS class whose profile names act like programming variables. The name of a RACFVARS profile can be specified as all or part of the names of general resource profiles that actually protect resources. The following sections describe ways in which you can make use of this facility.

- You can create a RACFVARS profile whose name is used as the entire profile name of profiles in the other class. On the ADDMEM operand used in creating the RACFVARS profile, you specify which resources are protected by the other profile.

  For example, a TAPEVOL profile name has only one qualifier, which is the tape volume being protected. You can create a RACFVARS profile named &TAPEV35, and specify several tape volumes on the ADDMEM operand. If a TAPEVOL profile has the same name as the RACFVARS profile (&TAPEV35), the TAPEVOL profile will protect those tape volumes.
- You can create a RACFVARS profile whose name is used as *part of* the profile name of profiles in the other class. The RACFVARS profile defines which values can be used for that part of the profile name.
- PERMIT commands issued for a RACFVARS profile affect the administration of that profile, not access to the resource protected with the RACFVARS name. For example, to allow users access to tape volumes protected by profile &TAPEV35 change the profile in the TAPEVOL class, not the profile in the RACFVARS class.

Using RACFVARS Profiles to Protect Many Resources

Like resource group classes (such as GTERMINL), you can use the RACFVARS class to create profiles that protect many resources that have *unlike* names.

To do this, take the following steps:

1. Create a profile in the RACFVARS class, specifying the resource names on the ADDMEM operand.

   ```
   RDEFINE  RACFVARS  profile-name  UACC(READ)
            ADDMEM(resource-name ...)
   ```

   where:

   profile-name
   must begin with the character &, can be up to 8 characters long (including the &) and cannot contain the characters *, %, or . (period). Profile names beginning with RAC are reserved for IBM use.

   UACC(READ)
   is specified to allow any user to use the RLIST command to determine how a particular variable is defined.

resource-name
can be up to 39 characters long and should not contain the characters *, %, or & because members containing those characters will not be effective.

For example, if you wish to use &PAYJOB as a qualifier in profile names in the JESSPOOL class, you could create a profile named &PAYJOB:

```
RDEFINE  RACFVARS  &PAYJOB  UACC(NONE)
         ADDMEM(TAXES CHECKS)
```

In any profile where you specify &PAYJOB, RACF uses TAXES and CHECKS for that qualifier.

2. Create another profile in another class with a profile name that includes the RACFVARS profile as part of its name. For convenience's sake, this profile is called the *protecting profile*, and its class is called the *protecting class*.

```
RDEFINE  protecting-class  profile-name  UACC(NONE)
```

Where part of profile-name has been defined in the RACFVARS profile defined in step "1" on page 107. For example:

```
RDEFINE  JESSPOOL  NODEA.*.&payjob;*.*.OUTPUT  UACC(NONE)
```

Note: These examples assume that a SETROPTS GENERIC(*classname*) was previously issued to turn generics on for this class and that a SETROPTS REFRESH was then done.

If the &PAYJOB profile contains members TAXES and CHECKS, then creating the JESSPOOL profile above is equivalent to:

```
RDEFINE  JESSPOOL  NODEA.*.TAXES.*.*.OUTPUT  UACC(NONE)
RDEFINE  JESSPOOL  NODEA.*.CHECKS.*.*.OUTPUT  UACC(NONE)
```

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:

. (period)
another &
%
*

For example, when using the RACFVARS class, you could enter:

```
RDEFINE RACFVARS &ABCDEFG ADDMEM(A B)
```

In this case, X.&ABCDEFGY.Z matches both X.AY.Z and X.BY.Z.

3. Permit users to the second profile, as needed:

```
PERMIT  profile-name  CLASS(protecting-class)
        ID(userid or group)  ACCESS(access-authority)
```

For example:

```
PERMIT  NODEA.*.&payjob;*.*.OUTPUT  CLASS(JESSPOOL)  UACC(NONE)
        ID(JOE TOM)  ACCESS(READ)
```

Note: Giving access authority to the RACFVARS profile does not affect access to the resources. (You might, however, give a user ALTER access authority to allow the user to change the RACFVARS profile.)

4. When you are ready to start using the protection defined in the RACFVARS profiles, activate the RACFVARS class and activate SETROPTS RACLIST processing for the RACFVARS class:

```
SETROPTS  CLASSACT(RACFVARS)  RACLIST(RACFVARS)
```

Note:

a. Activating the RACFVARS class may be required to allow certain other functions, such as JES spool reload, to work correctly.

b. If the RACFVARS class is already active, omit the CLASSACT operand.

c. Any time you change the value of a RACF variable (by specifying the ADDMEM or DELMEM operand on the RALTER command), you must refresh SETROPTS RACLIST processing for first, the RACFVARS class and, second, any class that uses the RACF variable.

d. See "Simplifying SURROGAT Administration for System Maintenance IDs" on page 81 for an example of using RACF variables to protect the use of LOGON BY with system maintenance IDs.

How RACF uses the RACFVARS member list

When RACF authorization checking matches a resource name with the name of a general resource profile that contains a RACF variable, it locates the RACFVARS profile that matches the RACF variable. RACF then compares each character of the resource name with each character of each member name in the RACFVARS profile until it finds the first match between a sequence of characters in the resource name and a RACFVARS member name. RACF compares the members in the order in which they occur in the RACFVARS profile. In other words, the first name in the member list is compared first and the last name is compared last until a match is found. When a match is found, RACF substitutes the member name for the RACF variable in the name of the general resource profile and then searches for a matching general resource profile to check the access authorization.

Administering the RACFVARS member list

Create the member list of a RACFVARS profile by issuing the ADDMEM operand of the RDEFINE command. When you specify multiple members, they are added to the RACFVARS profile in the *same* order that you specify them with the ADDMEM operand of the RDEFINE command. For example, if you specify `ADDMEM(A B)` with the RDEFINE command, the members are stored in the RACFVARS profile as `A B`.

If you issue the RALTER command to add one or more members to an existing RACFVARS profile, the new members are stored in the profile in the *reverse* of the order in which you specified them with the ADDMEM operand of the RALTER command. Additionally, if the existing profile already contains members, the new members are stored ahead of the existing members. For example, if you specify `ADDMEM(C D)` with the RALTER command to add members to an existing profile that already contains the members A B, the resulting member list stored in the profile is `D C A B`.

To view the members in a RACFVARS profile, issue the `RLIST RACFVARS` *`variable-name`* command. Note that the RLIST command lists the members in alphabetical order, not in the order in which they occur in the RACFVARS profile.

To view the members in the order in which they occur in a RACFVARS profile, use the output of the database unload (IRRDBU00) utility.

To reorder a RACFVARS member list, first use the RLIST command to list and make note of the members of the RACFVARS profile. Then delete the profile using the RDELETE command, and reissue the RDEFINE command with the ADDMEM operand to specify the members in the new order.

Because the order of the member names in the RACFVARS profile can be a critical factor in the successful matching of a resource name with the expected general resource profile, the following guidelines apply:

- When possible, avoid specifying a member name that is a subset of another member name in the same list. When impossible to avoid, the member name that is a subset of another name should follow the name of which it is a subset.
- Minimize the number of members in a single member list.
- Simplify the name of a general resource profile that contains a RACF variable by minimizing the use of generic characters after the initial ampersand (&) of the variable name.

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).
- 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. RACLISTing is accomplished using the SETROPTS command.

For example, the following profile:

```
RDEFINE  GTERMINL  DEPT35  UACC(NONE)
         ADDMEM(M01RF267 M03RF168 M04GG148)
```

Protects three terminals that have *unlike* names:

```
M01RF267
M03RF168
M04GG148
```

Table 17 on page 110 shows the resource group classes and their related member classes.

Table 17. Resource Group Classes

Resource	Resource Group Class	Related Member Class
Terminals	GTERMINL	TERMINAL
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. If you first activate generic profile checking for the related member class, you can include a generic character (* or % only) 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*. You must also activate SETROPTS RACLIST processing for the *member class*.

   For example, for terminals, issue the following command:

```
SETROPTS  CLASSACT(TERMINAL)  RACLIST(TERMINAL)
```

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
```

Adding a Resource to a Profile

To add a resource to a profile, issue the RALTER command with the ADDMEM operand, 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, 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 “Which Profile Is Used?” on page 111.) 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) on the RLIST command. The profiles that protect the terminal will appear in the RLIST output under "RESOURCE GROUPS".

For example, assume that the following commands were issued:

```
RDEFINE  GTERMINL  DEPT20  ADDMEM(T1 T2 T3)
RDEFINE  GTERMINL  DEPT22  ADDMEM(T3)
```

If you issue the following command:

```
RLIST  TERMINAL  T3  RESGROUP
```

the RLIST output will include the following:

```
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.

Which Profile Is Used?

If a resource is protected by more than one profile, all the profiles are checked and the following is done:

- The most restrictive UACC is used.
- For any particular user, the least restrictive of the access entries is used.

- The highest security level is used.
- Auditing is done if requested by any of the profiles.
- Category lists are combined.
- The first SECLABEL field found is chosen.

Note: The data supplied to the RACROUTE REQUEST=LIST exits contains flags to change the above rules if you desire. For more information, see *z/VM: RACF Security Server System Programmer's Guide*.

Recommendation: Do *not* specify the same resource in more than one profile.

Considerations for Resource Group Profiles

- Do not issue the SETROPTS RACLIST command for the resource group class (for example, GTERMINL). Instead, specify the related member class (for example, TERMINAL). When you RACLIST the TERMINAL class, RACF will RACLIST the GTERMINL class for you.
- 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 which generic profile produces a resource.
- A resource group profile is associated with one and only one resource class and cannot be used to group resources from two different classes.
- If you use resource grouping profiles, you should consider avoiding the use of the related member class. For example, if you use GTERMINL profiles, convert entirely to using GTERMINL profiles, then delete all TERMINAL profiles. This can ease the administration of terminal authorizations. For example, the SEARCH command will list profile names for only one class at a time: GTERMINL or TERMINAL.

  When converting generic TERMINAL profiles to GTERMINL profiles, you can specify generic characters on the ADDMEM operand to obtain the same coverage.

Activating and Refreshing Shared In-Storage Profiles for General Resources

RACF provides SETROPTS GENLIST processing and SETROPTS RACLIST processing to allow your installation to reduce the amount of storage used by general resource profiles and the amount of processing associated with retrieving profiles from the RACF database. The following sections describe these functions.

Note:

1. RACF does not allow you to specify SETROPTS GENLIST and SETROPTS RACLIST for the same general resource class.
2. If you RACLIST a class, global access checking is disabled for the class, because there is no performance benefit to checking the global access table compared with an in-storage profile.
3. Whenever you re-IPL the system, RACF automatically reactivates SETROPTS GENLIST and RACLIST processing for the classes for which this was previously requested.

SETROPTS Command Propagation

The options you specify on SETROPTS are common on systems that share the RACF database. All the systems involved must have the required levels of software. If you activate the SECLABEL and ML options on one system, they will be activated on all systems.

If you issue the SETROPTS command with any operand that changes the RACF database or issue the SETROPTS REFRESH command, the command is automatically propagated to all RACF servers that run on the same z/VM system, and to other systems in the same SSI cluster as the issuing system. Only the SETROPTS LIST command is not propagated.

On a system outside an SSI cluster, the action is not propagated to other systems that share the RACF database. You must issue the SETROPTS command separately for each system or restart the RACF servers on the other system or IPL the other system.

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 CDT in *z/VM: RACF Security Server Macros and Interfaces*.

To activate this function, issue the SETROPTS command with the GENLIST operand. 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 then 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)
```

If you have the SPECIAL attribute, you can also 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 class(es).

NOGENLIST is the default and is in effect for all eligible classes defined in the CDT when a RACF database is first initialized using IRRMIN00.

For more information about SETROPTS GENLIST processing, see *z/VM: RACF Security Server System Programmer's Guide*.

SETROPTS GENLIST Processing on Shared Systems

If your installation has two or more z/VM systems that share a RACF database, you only need to issue the SETROPTS GENLIST command on one of those systems. SETROPTS GENLIST processing will be automatically propagated to all systems that share the database and to every service machine in a multiple RACF service machine environment.

Refreshing Profiles for SETROPTS GENLIST Processing

If your installation has activated SETROPTS GENLIST processing for a particular resource class, you will need to refresh in-storage profiles for this processing when you make changes to one of these profiles. 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 class(es).

If you have the SPECIAL, AUDITOR, or OPERATIONS attribute at either the system or group level, 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.

The following example shows how to refresh in-storage generic profiles for the VMCMD and TERMINAL classes.

```
SETROPTS GENERIC(VMCMD TERMINAL) REFRESH
```

If you specify GENERIC(*), RACF refreshes profile lists for the DATASET class and all active classes in the class descriptor table except group resource classes.

When you initiate the refresh procedure, the refresh may not happen immediately. RACF refreshes a generic profile list the next time that the list is needed for an authorization check.

Note: The z/VM system does not use either the RACROUTE REQUEST=LIST or the RACROUTE REQUEST=FASTAUTH macro. However, a z/VM user can write an exit routine that invokes them.

If you specify NOGENERIC on the SETROPTS command, RACF stops using in-storage generic profile lists but does not immediately delete them. On z/VM, RACF deletes the profile lists only when you again specify GENERIC. When you specify GENERIC, RACF rebuilds the profile lists.

SETROPTS REFRESH Processing on Shared Systems
RACF does not automatically propagate the SETROPTS command to other non-cluster systems sharing the database for the combination of the GENERIC and REFRESH options. For this combination of options, the SETROPTS command must be issued to each system sharing the database. See “SETROPTS Command Propagation” on page 112.

SETROPTS RACLIST Processing

If you have the SPECIAL attribute, you can activate SETROPTS RACLIST processing. Activate this function when a general resource class contains a small number of frequently referenced profiles for which you cannot use global access checking. 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 CDT in *z/VM: RACF Security Server Macros and Interfaces*.

To activate this function, issue SETROPTS with the RACLIST operand. The following example shows how to activate SETROPTS RACLIST processing for the TERMINAL class.

```
SETROPTS  RACLIST(TERMINAL)
```

When you activate SETROPTS RACLIST processing for a general resource class, RACF copies both discrete and generic profiles for the specified class into common storage. 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 that profile. As a result, when you activate this function, you reduce processing overhead.

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)
```

Note: If you RACLIST a class, global access checking is disabled for the class, because there is no performance benefit to checking the global access table compared with an in-storage profile.

If you have the SPECIAL attribute, you can also deactivate SETROPTS RACLIST processing for general resource classes. To deactivate this option, issue SETROPTS with the NORACLIST operand and the selected general resource class(es).

NORACLIST is the default and is in effect for all eligible classes defined in the CDT when a RACF database is first initialized using IRRMIN00.

For more information about SETROPTS RACLIST processing, see *z/VM: RACF Security Server System Programmer's Guide*.

Note: SETROPTS RACLIST processing does not affect the use of the RACROUTE REQUEST=LIST macro. An installation can continue to code the RACROUTE REQUEST=LIST macro to use RACROUTE REQUEST=FASTAUTH for fast-path authorization checking.

SETROPTS RACLIST Processing on Shared Systems

RACF does not automatically propagate the SETROPTS command to non-cluster systems sharing the database for RACLIST option or the combination of the RACLIST and REFRESH options. For this combination of options, the SETROPTS command must be issued to each system sharing the database. See "SETROPTS Command Propagation" on page 112.

However, if you do not issue the SETROPTS command with the RACLIST option on a system sharing a RACF database and that system needs to re-IPL, RACLIST is performed for that system when a re-IPL occurs.

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 activate this option, issue SETROPTS with the RACLIST and REFRESH operands and specify the general resource class that contains the profiles you want to refresh. 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 TERMINAL classes.

```
SETROPTS RACLIST(TERMINAL) REFRESH
```

If you specify NORACLIST together with a general resource class on the SETROPTS command, you remove the profiles from storage for the class(es) that you specify.

Using SETROPTS STATISTICS for Statistics Collection

If you have the SPECIAL attribute, you can request that RACF record statistical information in discrete profiles for classes defined in the class descriptor table. 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. You specify this option with the STATISTICS operand of the SETROPTS command.

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.

The statistics that are recorded include:

- Date the resource was last referenced
- Date the resource was last updated (not recorded for terminals)
- Number of times the resource was accessed for each of the following access authorities: ALTER, CONTROL, UPDATE, and READ (only READ count is recorded for terminals)
- Number of times each user or group in the access list has accessed the resource.

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 has an impact on system performance, it is recommended that you deactivate this option until your installation evaluates the need to use it versus the potential performance impact. For more information, see *z/VM: RACF Security Server System Programmer's Guide*.

To disable statistics recording, use the NOSTATISTICS operand of the SETROPTS command.

STATISTICS Example

To help you understand how RACF maintains statistics, consider the following:

- USER1.191 is a minidisk profile.
- USER1.191 has a universal access (UACC) of READ.
- USER1 is in the access list with ALTER authority.
- 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 table.

If USER1 reads USER1.191, two counts are updated: the overall READ count, and the count in USER1's access list entry.

If USER2 reads the minidisk, two counts are updated: the overall READ count, and the count in USER2's access list entry.

If USER3 reads the minidisk, 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 minidisk, 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 minidisk, it gains access because of the universal access of READ, and the overall READ count increases.

If any user with OPERATIONS authority updates the minidisk, the overall UPDATE count increases.

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 the Creator's User ID to the Access List

The SETROPTS 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.

ADDCREATOR is the default unless IRRMIN00 is run with PARM=NEW.

Automatic Omission of the Creator's User ID from the Access List

The SETROPTS 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.

If IRRMIN00 is run with PARM=NEW, NOADDCREATOR is the default.

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 are printed elsewhere in this book for most of the kinds of resources supported by IBM-supplied general resource classes.

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 32.

       Note: To specify generic profile names, 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.
     - For most IBM-supplied classes, the rules for specifying profile names are described in this book.

     Notes:

     a. For some kinds of resources, such as terminals, 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 110 for more information.

     b. You can use RACFVARS profiles to specify which values will be taken by variables (indicated by an &) in profile names. See “Using RACF Variables in Profile Names (RACFVARS Class)” on page 107 for more information.
   - Decide which access is to be allowed to all users on the system who are not otherwise restricted. 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 118). 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 will be able to 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(s) protected by the profile. For more information, see step "11" on page 311 in "Authorizing Access to Resources Protected by RACF Profiles" on page 309.
  - 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 protected by this profile.
- Decide which user, if any, should be notified by a message when users make unsuccessful attempts to access resources protected by the profile (NOTIFY operand).
- Decide whether RACF should log access attempts to resources protected by the profile (AUDIT operand).

  Note: To see the results of the logging done by RACF, use the RACF SMF data unload utility or the RACF report writer. For more information, see *z/VM: RACF Security Server 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 specific fields for which you should assign values. For example:
  - Profiles in the APPCLU class have SESSION segments.
  - Profiles in the TAPEVOL class have the SINGLEDS 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.

  For specific information on these operands, see the appropriate section of this book, or see the description of the RDEFINE command in *z/VM: RACF Security Server Command Language Reference*.
- 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  class-name  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 profile, 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(class-name)
       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(class-name)
       ID(userid or group) ACCESS(access-authority)
       WHEN(condition)
```

Note:

a. Access authorities 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. This book does not describe all classes. For descriptions of classes, see the class descriptor table (CDT) in *z/VM: RACF Security Server Macros and Interfaces*. Also, for some classes (the FACILITY class, for example), the access required by some resource managers to specific profiles is described in the documentation of those resource managers.

4. If you have not already done so, activate the resource class:

```
SETROPTS  CLASSACT(class-name)
```

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 (GAC) entry that has a name similar to the new resource profile.

  See Chapter 7, “Fast Authorization Using the Global Access Checking (GAC) Table,” on page 121.

- Reduce I/O to the RACF database by requesting that RACF keep all profiles in the class in storage:

```
SETROPTS  RACLIST(class-name)
```

  Note: This is required for some classes.

- On z/VM, where large numbers of profiles can consume too much system storage, keep only generic profiles (not discrete profiles) in storage:

```
SETROPTS  GENLIST(class-name)
```

Chapter 7. Fast Authorization Using the Global Access Checking (GAC) Table

You can use global access checking to improve 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 give all users on the system READ access to the MAINT 190 minidisk.

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.

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, and SETROPTS RACLIST processing is not in effect for the class, 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.

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 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

In planning the resources to be *public* through global access checking, consider the following:

- MAINT 190 system disk (CMS S-disk)
- MAINT 19E system disk extension (CMS Y-disk)
- ISPF minidisks
- Local system extension disks
- Disks for widely-used licensed program, such as APL
- Tools minidisks
- Virtual unit record devices for the RSCS service machine (VMRDR class)
- Frequently used RSCS nodes (VMNODE class)

For more information on protecting public minidisks, see *z/VM: RACF Security Server System Programmer's Guide*.

Note: Even greater performance benefit can be achieved for minidisk access checking by using the global disk table in CP. See *z/VM: RACF Security Server Macros and Interfaces* for GLBLDSK macro documentation.

Creating Global Access Checking Table Entries

To create an entry in the global access checking table, do the following:

1. Plan the entries for the global access checking table, using the following guidelines:
   a. Attempt to identify resource profiles that are accessed frequently and for which a performance benefit is desired.
   b. Do *not* add entries to the global access checking table for profiles in classes that are RACLISTed. RACF will not search the global access checking table for profiles in RACLISTed classes.
   c. 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 minidisk profile PROFS.399 with UACC(READ) and AUDIT(FAILURES(UPDATE)) specified, you might create a global access checking table entry for it as follows:

      ```
      RALTER GLOBAL VMMDISK ADDMEM(PROFS.399/READ)
      ```

      However, if there are users or groups in the standard access list of profile PROFS.399 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 PROFS minidisk.
   d. If you have resources protected by a generic profile with UACC other than NONE, and others 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 MAINT.* with a UACC(READ), but you also have some specific profiles with more restrictive entries, such as, MAINT.191 with UACC(READ) and an access list with JOE/NONE, then create two entries:

      ```
      MAINT.191/NONE
      MAINT.*/READ
      ```

      The entry with /NONE will not fail any attempts but will stop requests for MAINT.191 from being granted by the MAINT.* 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  class-name
   ```
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 class-name 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 *class-name* (this is done with 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).

Once added, generic entries in the global access checking table are used in global access checking even if generic profile checking is turned off (SETROPTS NOGENERIC command).

The resource name can also include the name qualifiers &RACUID (RACF user ID) or &RACGPID (*current connect group*—see Glossary). For example, the following entry allows users to have ALTER access to minidisks that begin with their own user IDs:

The resource name can also include variables defined in the RACFVARS class.

```
RALTER GLOBAL VMMDISK ADDMEM(&RACUID.*/ALTER)
```

Note: The preceding entry does not *change* a user's access to his or her own minidisks, 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.

See *z/VM: RACF Security Server Command Language Reference* for more information on specifying the ADDMEM option.

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(class-name)
```

Attention:
Save a listing of the global access checking table. This can assist you in recovering 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. It is recommended that you write an EXEC containing the commands used 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. It is strongly recommended 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  class-name  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  class-name  ADDMEM(resource-name/level)
SETROPTS  GLOBAL(class-name)  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  class-name  DELMEM(resource-name/level)
SETROPTS  GLOBAL(class-name) REFRESH
```

Attention:
Do not use the RDELETE command unless you intend to delete the entire global access checking table for that class.

Stopping Global Access Checking for a Specific Class

To stop global access checking for a specific class, issue the following command:

```
SETROPTS  NOGLOBAL(class-name)
```

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:

  ```
  RLIST  GLOBAL  class-name
  ```

  This shows the entries in the order in which they are searched by RACF.
- For a list that shows all entries in the table, see the DSMON report that lists the global access checking table (described in *z/VM: RACF Security Server Auditor's Guide*).

Refreshing Global Access Checking Lists

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. The following example specifies that you want RACF to refresh global access checking lists for the VMCMD and TERMINAL classes.

```
SETROPTS GLOBAL(VMCMD TERMINAL) REFRESH
```

If you specify GLOBAL(*), RACF refreshes access checking lists for the DATASET class and all active classes in the class descriptor table.

If you specify NOGLOBAL, you disable global access checking for the class that you specify.

SETROPTS REFRESH Processing on Shared Systems
RACF does not automatically propagate the SETROPTS command to other non-cluster systems sharing the database for the combination of the GENERIC and REFRESH options. For this combination of options, the SETROPTS command must be issued to each system sharing the database. See "SETROPTS Command Propagation" on page 112. However, if you do not perform a refresh (issue the SETROPTS command with the REFRESH option) on a system sharing a RACF database and that system needs to re-IPL, the refresh takes effect on that system when re-IPL is performed.

Examples for z/VM Systems

Example 1: MAINT Minidisks

To allow all users to have READ access to all minidisks owned by MAINT except the MAINT 191 minidisk, do the following:

```
    SETROPTS  GLOBAL(VMMDISK)

    RDEFINE  GLOBAL  VMMDISK

(1) RALTER  GLOBAL  VMMDISK  ADDMEM(MAINT.*/READ)
(2) RALTER  GLOBAL  VMMDISK  ADDMEM(MAINT.191/NONE)

    SETROPTS  GLOBAL(VMMDISK)  REFRESH

(3) RDEFINE  VMMDISK     MAINT.*    UACC(READ)
(4) PERMIT   MAINT.*    CLASS(VMMDISK)  ID(SYSGROUP)  ACCESS(CONTROL)

(5) RDEFINE  VMMDISK     MAINT.191  UACC(NONE)
(6) PERMIT   MAINT.191  CLASS(VMMDISK)  ID(SYSGROUP)  ACCESS(CONTROL)
(7) SETROPTS GENLIST(VMMDISK)
```

The entry defined at (1) has the same name and access authority (READ) as the profile created at (3).

The entry defined at (2) has an access authority of NONE, which causes RACF to use a profile in the VMMDISK class. In this case, RACF uses the profile created at (5).

The access lists created at (4) and (6) allow SYSGROUP a higher authority (CONTROL) than the global access table entries (READ and NONE).

The command at (7) should be used if you are using generic profiles.

The following command would have no effect on USER1's attempts to LINK to the MAINT 190 disk, because the entry created at (1) protects this minidisk, and allows everyone READ access without checking a profile:

```
PERMIT   MAINT.190  CLASS(VMMDISK)  ID(USER1)  ACCESS(NONE)
```

To make such a PERMIT effective, you must also do the following:

```
RALTER  GLOBAL  VMMDISK  ADDMEM(MAINT.190/NONE)
```

Example 2: SFS Files and Directories

To allow all users to have ALTER access to their own SFS files and directories, do the following:

```
SETROPTS GLOBAL(FILE DIRECTRY) GENERIC(FILE DIRECTRY)
RDEFINE GLOBAL (FILE DIRECTRY)
RALTER GLOBAL DIRECTRY ADDMEM(*.&racuid.**/ALTER)
RALTER GLOBAL FILE ADDMEM(*.&racuid.**.*.*/ALTER)
```

To allow all users to have READ access to an SFS directory named POOL1:TOOLS.GENUSER and to the files within the directory, do the following:

```
RALTER GLOBAL DIRECTRY ADDMEM(POOL1.TOOLS.GENUSER/READ)
RALTER GLOBAL FILE ADDMEM(POOL1.TOOLS.GENUSER.*.*/READ)
```

Note: You must use the RACF format of SFS names for defining entries in the global access table. See For more information, see "RACF Format for SFS Directory and File Names" on page 173.

Example 3: The VMBATCH Class

If the VMBATCH class is active, you should consider specifying the following:

```
RALTER GLOBAL VMMDISK ADDMEM(&RACUID.*/ALTER)
```

This allows batch machines to access the minidisks of any user who submits batch requests. For more information, see "Protecting Alternate User IDs" on page 159.

Special Considerations for Global Access Checking

When using global access checking, consider the following:

- While RACF is searching the global access checking table for a matching entry, profiles in the class are ignored. 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.
- 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, 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 or RACHECK macro).
- Updated global access checking table entries become effective with the next IPL or after execution of the SETROPTS command with the GLOBAL(*class-name*) 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 assure that auditing or security classification checking will take place for a specific data set.

Chapter 8. Protecting z/VM with VMXEVENT Profiles

When RACF is installed, the z/VM operating system calls RACF to provide security protection beyond the operating system's basic features. This chapter discusses how you can use RACF to meet your operating system security needs. Previous chapters provided an overview to assist you in understanding your security needs and defining an installation security policy. Once you understand your security policy, use this chapter to begin implementing resource protection.

The z/VM control program (CP) defines the z/VM events that require authorization checking by RACF. These are referred to as controllable events. All controllable event are, by default, controlled. The security administrator determines whether the default controls meet the requirements of the installation's security policy. Controls can be turned on or off by using profiles in the VMXEVENT class.

CP always calls RACF for authorization of the LOGON, AUTOLOG, and XAUTOLOG events. This control cannot be turned off.

Control can be established on a system-wide or individual user basis. Details on defining VMXEVENT profiles is described in “System z/VM Event Profiles” on page 130 and “Individual z/VM Event Profiles” on page 131.

Although all controllable events can be audited, none are audited by default. Events can be audited through use of VMXEVENT profiles. For a complete discussion on auditing, see *z/VM: RACF Security Server Auditor's Guide*.

The security administrator implements the installation's security policy by using RACF commands to:

- Define profiles in a general resource class
- Permit users to the resources within the class
- Activate the class
- Make sure the CP event mediating access to the resource type is being controlled

The z/VM operating system may call RACF for authorization checking of certain other z/VM events. Your security policy may not find these calls necessary. You can use RACF commands to turn control on or off for these z/VM events.

The z/VM operating system determines the z/VM events that require authorization checking by RACF. These are referred to as *controllable* z/VM events. By default, control is on for certain z/VM events. The security administrator determines whether the default controls are needed for implementing the installation's security policy as control can be turned off by using profiles in the VMXEVENT class.

The z/VM operating system also provides auditable z/VM events. By default, there is no auditing done on any z/VM events. For a complete discussion on auditing, see *z/VM: RACF Security Server Auditor's Guide*.

Table 18 on page 127 shows the relationship of resources to protect, the z/VM name of the resource (z/VM event), and the RACF class associated with the resource.

Table 18. z/VM Events, Resources, and RACF Classes

z/VM Event	Resource to Protect	RACF Class	Chapter Reference
APPCPWVL	APPC connection with password	USER	“Using the APPCPWVL Event to Verify Passwords on APPC CONNECT” on page 149
AUTOLOG	System access	USER	“Individual z/VM Event Profiles” on page 131
COUPLE.G	Connection to a Guest LAN	VMLAN	“Controllable z/VM Events” on page 134
DIAG088	Use of Diagnose Code X'88'	VMCMD and SURROGAT	“Protecting the DIAGNOSE X'88' Subcodes” on page 147

Table 18. z/VM Events, Resources, and RACF Classes (continued)

z/VM Event	Resource to Protect	RACF Class	Chapter Reference
DIAG0A0	Use of Diagnose Code X'A0'	VMCMD	"Protecting the DIAGNOSE X'A0' Subcodes" on page 148
DIAG0D4	Alternate User ID	VMBATCH	"Protecting Alternate User IDs" on page 159
DIAG0E4	Use of Diagnose Code X'E4'	VMCMD	"Protecting the STORE.C, TRSOURCE, and DIAG0E4 Events" on page 143
DIAG280	BFS set-UID and set-GID files	VMPOSIX	"Protecting Set-UID and Set-GID Executable Files" on page 205
DEFINE.MDISK	DASDs and MDISKs	VMCMD	"Protecting the DEFINE.MDISK Command" on page 144
FOR	target user of command	SURROGAT	"Protecting the FOR Command" on page 145
LINK	Links to other's minidisks	VMMDISK	"Protecting z/VM Minidisks" on page 152
LOGON	System access	USER	"Individual z/VM Event Profiles" on page 131
	Terminals	TERMINAL GTERMINL	
MDISK	Links to own minidisks	VMMDISK	"Protecting z/VM Minidisks" on page 152
RSTDSEG	Restricted segments (NSS or DCSS)	VMSEGMT	"Protecting Restricted Shared Memory Segments" on page 161
RDEVCTRL	Real Devices	VMDEV	"Protecting Real Devices" on page 151
STORE.C	Use of STORE HOST command	VMCMD	"Protecting the STORE.C, TRSOURCE, and DIAG0E4 Events" on page 143
TAG	RSCS nodes	VMNODE	"Protecting RSCS Nodes" on page 157
TRANSFER.D	Virtual unit record devices	VMRDR	"Protecting Virtual Unit Record Devices" on page 156
TRANSFER.G	Virtual unit record devices	VMRDR	"Protecting Virtual Unit Record Devices" on page 156
TRSOURCE	Use of the TRSOURCE command	VMCMD	"Protecting the STORE.C, TRSOURCE, and DIAG0E4 Events" on page 143
XAUTOLOG (AB version)	System access	USER	"Protecting XAUTOLOG ON" on page 145
	Ability to use the XAUTOLOG ON parameter	VMCMD	
XAUTOLOG (G version)	System access	USER	"Protecting the XAUTOLOG.G Command" on page 144
	Ability to XAUTOLOG someone else	VMCMD	

For more information about BFS set-UID and set-GID files, see "Protecting Set-UID and Set-GID Executable Files" on page 205.

For z/VM, the following events are controlled by default:

APPCPWVL
COUPLE.G
DIAG088
DIAG0A0
DIAG0D4
DIAG0E4

DIAG280
FOR.C
FOR.G
LINK
MDISK
RDEVCTRL
RSTDSEG
STORE.C
TAG
TRANSFER.D
TRANSFER.G
TRSOURCE
XAUTOLOG.ON

These events, along with their control settings, are listed in the SETEVENT LIST output. If your installation changed the default settings, the changes will be reflected in the SETEVENT LIST output. Sample SETEVENT LIST output is in Figure 7 on page 136.

If control for an event is off, RACF does not perform authorization checking for the general resource class because z/VM does not call RACF. If control for an event is on, RACF performs authorization checking for the general resource class relevant to the particular event.

For example, LINK is protected by profiles in the general resource class VMMDISK. LINK is also a controllable z/VM event. If the VMXEVENT profile indicates authorization checking by RACF (control is on), then the VMMDISK profiles determine if authorization is granted. If the VMXEVENT profile indicates no authorization checking by RACF (control is off), then z/VM security determines if authorization is granted.

Creating VMXEVENT Profiles

If the default z/VM event settings do not meet your security needs, you must create a VMXEVENT profile for your installation.

When creating VMXEVENT profiles, remember that they serve a dual purpose. They can be used to instruct z/VM to call RACF to perform *access checking* on designated z/VM events and they can be used to instruct z/VM to call RACF to perform *auditing* on designated z/VM events.

The ability to create these various types of profiles depends on the attribute of the user creating the profile:

- A user with the SPECIAL attribute can define profiles in the VMXEVENT class, but can *only* set the *control* options for z/VM events in that profile.
- A user with the AUDITOR attribute and CLAUTH to the VMXEVENT class can define profiles in the VMXEVENT class, but can *only* set the *audit* options for z/VM events in that profile.

You can use one profile to define both z/VM auditing and access calls to RACF. Alternatively, you can use one profile to indicate that z/VM should call RACF to audit certain events and another profile to indicate that z/VM should call RACF to perform access checking on certain events.

If the roles of auditor and RACF administrator are separated (no user has both AUDITOR and SPECIAL), then separate VMXEVENT profiles *may* be used to change audit and control settings. If the roles are combined (the user has both AUDITOR and SPECIAL), then *both* settings *must* be kept in a single VMXEVENT profile. Use of separate profiles in this latter case will result in audit or control settings being reset back to their default values.

If the SPECIAL user wants the profile to also contain z/VM events to be audited, the SPECIAL user must place the user with the AUDITOR attribute on the access list of the VMXEVENT profile with an access of ALTER.

To easily adapt to changes in auditing or access control requirements, you might want to define several VMXEVENT profiles to use as your auditing or access control environment changes.

To activate the VMXEVENT resource class, issue the following command:

```
SETROPTS CLASSACT(VMXEVENT)
```

You are now ready to define your profiles.

System z/VM Event Profiles

A system z/VM event profile is a resource profile defined in the VMXEVENT class. The options set in a system z/VM event profile determine the type of auditing and control that will take place for all of the users on the system. The rest of this section discusses how to use a system z/VM event profile to control and not control z/VM events. For information specific to auditing z/VM events, see *z/VM: RACF Security Server Auditor's Guide*.

If an individual z/VM event profile is present for any specific user, it takes precedence over a system z/VM event profile (see "Individual z/VM Event Profiles" on page 131).

Creating a System z/VM Event Profile

Use the RDEFINE command to create a system z/VM event profile. For example, the following command defines a VMXEVENT profile called EVENTS1.

```
RDEFINE VMXEVENT EVENTS1
```

Altering a System z/VM Event Profile to Stop RACF Authorization Checking

When RACF is first installed, the default is that all controllable events are controlled. This means that z/VM always calls RACF for authorization checking for these events. If you want RACF to perform authorization checking for these events, you do not need to make any changes. You can use the RALTER command to add a member to a VMXEVENT profile for each event for which you wish to turn off control.

For example, the following command specifies that, when profile EVENTS1 is in effect on the system, z/VM does not call RACF for authorization checking for the LINK and TAG commands.

```
RALTER VMXEVENT EVENTS1 ADDMEM(LINK/NOCTL TAG/NOCTL)
```

The first part of the member name (z/VM event) must match exactly the "z/VM EVENT" shown in the "CONTROLLABLE z/VM EVENTS" section of the output of the SETEVENT LIST command. For a sample of this output, see Figure 7 on page 136.

You can issue the RALTER command as many times as you need for one VMXEVENT profile, adding or deleting members as necessary.

Note:

1. You can combine creating and altering the profile by specifying the ADDMEM operand on the RDEFINE command.
2. If you use the RACF ISPF panels to update a VMXEVENT profile, you can select z/VM event names from a list on the panel.
3. When you issue the RLIST command for a VMXEVENT profile, the output shows the members that have been added to the profile.

Activating a System z/VM Event Profile

Use the SETEVENT command to specify which VMXEVENT profile you want active. Depending on the access control requirements of your environment at a given time, various profiles are appropriate to meet those requirements. For example, the following command activates the access checking set in the profile EVENTS1.

```
SETEVENT REFRESH EVENTS1
```

When a profile is active, any changes you make to it do not take effect until a subsequent SETEVENT REFRESH command is issued. The currently-active profiles are displayed in the SETEVENT LIST output.

Note: If you are sharing a database between two or more z/VM systems, you only need to issue the SETEVENT REFRESH command on one system.

Removing z/VM Events from a System z/VM Event Profile

Use the RALTER command to remove a member from a profile for each event that is to be returned to its default setting. For example, the following command deletes the settings for the LINK and TAG events which were created in a previous example.

```
RALTER VMXEVENT EVENTS1 DELMEM(LINK/NOCTL TAG/NOCTL)
```

The first part of the member name (z/VM event) must match exactly the "z/VM EVENT" shown in the "CONTROLLABLE z/VM EVENTS" section of the output of the SETEVENT LIST command. For a sample of this output, see Figure 7 on page 136.

For these changes to take effect, the EVENTS1 profile must be reactivated by the SETEVENT REFRESH command, as described in “Activating a System z/VM Event Profile” on page 130. Once the SETEVENT REFRESH has been issued, the settings for the LINK and TAG events will return to the default of being controlled.

Individual z/VM Event Profiles

An individual z/VM event profile is a resource profile defined in the VMXEVENT class. The options set in an individual z/VM event profile determine the type of auditing and control that will take place for the user. If present, an individual z/VM event profile takes precedence over a system z/VM event profile in determining when z/VM calls RACF.

The main objective in using an individual z/VM event profile is to identify users on the system who have unique circumstances in regard to auditing and access control, and to tailor selective profiles to monitor them in a specific way, which may result in either more or less monitoring for these users than for other users on the system.

Suppose, for instance, that a user issues a particular command or series of commands frequently throughout the day. The installation could turn off RACF security checking for these commands by creating an individual z/VM event profile for the user. As a result of reduced RACF calls, performance may be improved.

Creating an Individual z/VM Event Profile

Use the RDEFINE command to create an individual z/VM event profile. Individual z/VM event profiles are distinguished from system z/VM event profiles by a high-level qualifier called USERSEL. To identify the profile you are creating as an individual z/VM event profile, you must specify the high-level qualifier, followed by the user's user ID.

For example, if you wanted to create an individual z/VM event profile for a user with the user ID of FRANK, enter the following command:

```
RDEFINE VMXEVENT USERSEL.FRANK
```

Note:

1. It is possible to define an individual z/VM event profile with no auditing specified, and no events controlled, although you should be aware of the implications. See “Per-User Overrides of RACF Control and Audit Settings” on page 133 for more information.
2. Only one individual z/VM event profile is allowed to be defined for each user; therefore, both control and auditing options must be specified in the same profile.

The profile is put into effect the next time the user ID is logged on, is autologged, or reconnects. To activate the profile while the user ID is logged on, use the SETEVENT command as follows:

```
SETEVENT REFRESH USERSEL.FRANK
```

Altering an Individual z/VM Event Profile to Stop Authorization Checking

When an individual z/VM event profile containing no members is activated, the default is that all controllable events are controlled for the user and no events are audited. (To see how to activate auditing for z/VM events for an individual user, see *z/VM: RACF Security Server Auditor's Guide*.) Note that this is true regardless of what is specified in the system z/VM event profile. If you do not want RACF to perform authorization checking for specific events for the user, you can use the RALTER command to add a member to the individual z/VM event profile for each event for which you wish to turn off control.

For example, the following command specifies that when profile USERSEL.FRANK is in effect for user FRANK, z/VM does not call RACF each time user ID FRANK issues the TAG and the LINK commands.

```
RALTER VMXEVENT USERSEL.FRANK ADDMEM(TAG/NOCTL LINK/NOCTL)
```

The first part of the member name (z/VM event) must match exactly the "z/VM EVENT" shown in the "CONTROLLABLE z/VM EVENTS" section of the output of the SETEVENT LIST command. See Figure 7 on page 136.

You can issue the RALTER command as many times as you need for one VMXEVENT profile, adding or deleting members as necessary.

Note:

1. You can combine creating and altering the profile by specifying the ADDMEM operand on the RDEFINE command.
2. If you use the RACF ISPF panels to update a VMXEVENT profile, you can select z/VM events from a list on the panel.
3. The options set in this profile do not take effect until SETEVENT REFRESH USERSEL.FRANK has been issued or until user ID FRANK logs on again, is autologged, or reconnects.
4. When you issue the RLIST command for a VMXEVENT profile, the output shows the members that have been added to the profile.

Activating an Individual z/VM Event Profile

An individual z/VM event profile will be activated or refreshed automatically whenever the user logs on, is autologged, or reconnects. You may also refresh the profile while the user is currently logged on by using the SETEVENT REFRESH command. For example, the following command resets the control and auditing options for user FRANK:

```
SETEVENT REFRESH USERSEL.FRANK
```

Note that the user FRANK must be logged on when the SETEVENT REFRESH is issued, or an error message will be displayed.

Suspending an Individual z/VM Event Profile

Use the SETEVENT RESET command to return a user to the auditing and authorization checking set in a system z/VM event profile. For example, to discontinue individual auditing and authorization checking for user ID FRANK, enter the following command:

```
SETEVENT RESET USERSEL.FRANK
```

Note: The SETEVENT RESET command does not delete the individual z/VM event profile for user ID FRANK. Issuing the command simply means that you *temporarily suspend* the use of the individual z/VM event profile that has been established for user ID FRANK. The suspension will stay in effect until you issue a REFRESH for user ID FRANK's individual z/VM event profile or until user ID FRANK next logs on, is autologged, or reconnects.

Deleting an Individual z/VM Event Profile

Use the RDELETE and SETEVENT RESET commands to not merely suspend, but to *delete* an individual z/VM event profile. For example, to delete user ID FRANK's individual z/VM event profile and have user ID FRANK be subject to the system z/VM event profile that is in effect on the system, follow this sequence. First, delete the user's individual z/VM event profile to ensure that the user's individual z/VM event profile will not be reactivated at LOGON. Enter the following command:

```
RDELETE VMXEVENT USERSEL.FRANK
```

Second, issue the SETEVENT RESET command to deactivate the user's individual z/VM event profile. The high-level qualifier is required when issuing this command, even though the previous command has in fact removed the individual z/VM event profile.

```
SETEVENT RESET USERSEL.FRANK
```

You would use this same sequence to delete the exempt status of a user by removing the user's individual z/VM event profile, thus making that user subject to the system z/VM event profile.

Note: You can specify RDELETE without specifying SETEVENT RESET and thus allow the user to be under the options of the user's individual z/VM event profile until the user logs off. However, if the user simply disconnects, and does not logoff, auditing and authorization checking for the user will continue to be done through the individual z/VM event profile. The safer course of action, if you want to remove the use of an individual z/VM event profile, is to issue the RDELETE and SETEVENT RESET commands in sequence.

Per-User Overrides of RACF Control and Audit Settings

An installation may exempt a particular user from particular command controls or auditing; however, the installation must be aware of the implications. To exempt a user, the installation creates an individual z/VM event profile with a member list that specifies that all controllable z/VM events are not to be controlled (do not require authorization checking) and the audit options are not set for any z/VM events. For an accurate and up-to-date list of controllable z/VM events, issue the SETEVENT LIST command.

When the exempt user ID enters the system (by means of the LOGON, AUTOLOG or XAUTOLOG command), RACF will recognize that nothing is being audited or controlled for this user; at this point z/VM will no longer call RACF for authorization checking during the exempt user's course of operation. As a result of reduced RACF calls, performance may be improved. However, the user ID will still be subject to CP directory authorization, as if RACF were not installed.

When subsequent CP commands are processed for the exempt user ID, z/VM will not invoke RACF for any AUDIT, MAC, or DAC requests, except for a small subset of events for which RACF must be invoked (AUTOLOG, LOGON, LOGOFF, and XAUTOLOG).

Attention:
Exempt profiles may be created for any user, at the installation's discretion. Keep in mind, however, that if an exempt profile is created for a service machine that does work on behalf of other users (as, for example, a batch machine), then jobs submitted to that machine will be executed with virtually no authorization checking or auditing being performed by RACF.

For example, to define a VMXEVENT profile for the exempt user ID named SERVER1, use the RDEFINE command as follows:

```
RDEFINE VMXEVENT USERSEL.SERVER1
```

This will create a profile which by default contains no audited members and for which control is on for all controllable events. For the controllable events, members must be added explicitly to disable control for these events:

```
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(LINK/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(COUPLE.G/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(STORE.C/NOCTL)
```

```
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(TAG/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(TRANSFER.D/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(TRANSFER.G/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(TRSOURCE/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(DIAG088/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(FOR.C/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(FOR.G/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(DIAG0D4/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(DIAG0E4/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(APPCPWVL/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(MDISK/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(RDEVCTRL/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(RSTDSEG/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(DIAG0A0/NOCTL)
RALTER VMXEVENT USERSEL.SERVER1 ADDMEM(DIAG280/NOCTL)
```

Notes:

1. This list can change because of product updates. For an accurate and up-to-date list of controllable z/VM events, issue the SETEVENT LIST command.
2. A REXX exec can be created to define these profiles for several exempt user IDs.

The profile is put into effect the next time the user ID is logged on, is autologged, or reconnects. To activate the profile while the user ID is logged on, use the SETEVENT command. For example:

```
SETEVENT REFRESH USERSEL.SERVER1
```

Considerations for User IDs Autologged During System Initialization

Certain user IDs are logged on during system initialization. These user IDs include the following:

- AUTOLOG1— Logged on automatically at IPL time to perform functions you select
- OPERACCT— For CP *ACCOUNT system service to accumulate and checkpoint accounting records
- OPEREREP— For CP *LOGREC system service to accumulate and checkpoint error records
- OPERSYMP— For CP *SYMPTOM system service to accumulate and checkpoint symptom records
- OPERATOR— Primary system operator's user ID.

The autologged user IDs are tailorable using a system configuration file called SYSTEM CONFIG. RACF is not active on the system until after these user IDs have been logged on during system initialization. At this point, RACF has not been called to authenticate them; therefore, individual z/VM event profiles are not in effect for any of them, with the exception of the OPERATOR user ID. RACF does activate an individual z/VM event profile for the system operator, but not for the others. If your installation has an individual z/VM event profile for a user autologged during system IPL (besides OPERATOR), a SETEVENT REFRESH must be issued to activate the profile.

Because RACF autologs the user ID, AUTOLOG2, this SETEVENT command could be added to the PROFILE EXEC of AUTOLOG2. For example, if your installation has an individual z/VM event profile for the OPERACCT user, the following statement can be added to the PROFILE EXEC of AUTOLOG2:

```
RAC SETEVENT REFRESH USERSEL.OPERACCT
```

To authorize the use of the SETEVENT command for AUTOLOG2, you must ensure the AUTOLOG2 user ID has access to the RAC command processor and has the RACF SPECIAL and/or AUDITOR attribute, depending on what settings are being activated in the individual z/VM event profile.

Until RACF is active, all z/VM requests (z/VM events) from these autologged users are exempt from security authorization and auditing. Once RACF becomes active, z/VM requests from these users are handled using authorization and auditing rules which are in effect for that user or all users on the system.

For more information on SYSTEM CONFIG, see *z/VM Planning and Administration.*

Controllable z/VM Events

You can use the SETEVENT LIST command to generate a list of controllable and auditable z/VM events.

In the SETEVENT LIST output, examine the "CONTROLLABLE z/VM EVENTS" section of the output. (z/VM events listed in the "AUDITABLE z/VM EVENTS" section are not necessarily controllable.)

In the SETEVENT LIST output, "z/VM EVENT" indicates the name of the z/VM event as RACF recognizes it. "STATUS" indicates one of the following:

- NO_CONTROL indicates that z/VM is not to call RACF when the z/VM event occurs. This means that any user whose z/VM privilege class allows the use of the event in question can issue the command or code without checking a RACF resource profile.
- CONTROL indicates that z/VM is to call RACF when the z/VM event occurs. z/VM may or may not enforce its own authorization mechanisms in addition (for example, privilege class and/or directory authorization), depending on the event.

Normal RACF authorization considerations, such as the use of security labels, can affect the granting of authority when these profiles are checked.

Note: If the command is prefixed with "RAC", the output of the command is written to a file on your disk or directory accessed as A. You can print this file for study, and edit it to generate the appropriate RACF commands to protect selected z/VM events.

Commands issued before logon (such as DIAL, UNDIAL, and MESSAGE) have no user ID associated with them. To prevent this, see “Preventing Use of DIAL, UNDIAL, and MESSAGE Commands Before Logon” on page 142.

Sample output from the SETEVENT LIST command is shown in Figure 7 on page 136.

```
PRE-LOGON COMMANDS

COMMAND              CONFIGURED IN
-------              -------------
DIAL                   YES
MESSAGE.ANY            YES
UNDIAL                 YES

CONTROLLABLE VM EVENTS       CURRENT SYSTEM CONTROL PROFILE NAME: EVENTS1

VM EVENT                     STATUS     VM EVENT                     STATUS
--------                     ------     --------                     ------
COUPLE.G                     CONTROL    FOR.C                        CONTROL
FOR.G                        CONTROL    LINK                         CONTROL
STORE.C                      CONTROL    TAG                          CONTROL
TRANSFER.D                   CONTROL    TRANSFER.G                   CONTROL
TRSOURCE                     CONTROL    DIAG088                      CONTROL
DIAG0A0                      CONTROL    DIAG0D4                      CONTROL
DIAG0E4                      CONTROL    DIAG280                      CONTROL
APPCPWVL                     CONTROL    MDISK                        CONTROL
RSTDSEG                      CONTROL    RDEVCTRL                     CONTROL


AUDITABLE VM EVENTS          CURRENT SYSTEM AUDIT PROFILE NAME: EVENTS1

VM EVENT                     STATUS     VM EVENT                     STATUS
--------                     ------     --------                     ------
ACNT                         NO_AUDIT   ACTIVATE                     NO_AUDIT
ADJUNCT                      NO_AUDIT   ASSOCIATE                    NO_AUDIT
AT                           NO_AUDIT   ATTACH                       NO_AUDIT
ATTN                         NO_AUDIT   AUTOLOG.A                    NO_AUDIT
AUTOLOG.B                    NO_AUDIT   BACKSPACE                    NO_AUDIT
BEGIN                        NO_AUDIT   CHANGE.D                     NO_AUDIT
CHANGE.G                     NO_AUDIT   CLOSE                        NO_AUDIT
COMMANDS                     NO_AUDIT   COMMIT                       NO_AUDIT
CONCOPY                      NO_AUDIT   COUPLE.G                     NO_AUDIT
CPACCESS                     NO_AUDIT   CPCACHE                     NO_AUDIT
CPHX                         NO_AUDIT   CPLISTFILE                  NO_AUDIT
CPRELEASE                    NO_AUDIT   CPTYPE                      NO_AUDIT
CPU                          NO_AUDIT   CPVLOAD                     NO_AUDIT
CPXLOAD                      NO_AUDIT   CPXUNLOAD                   NO_AUDIT
DEACTIVE                     NO_AUDIT   DEACTIVATE                  NO_AUDIT
DEFINE.A                     NO_AUDIT   DEFINE.B                    NO_AUDIT
DEFINE.E                     NO_AUDIT   DEFINE.G                    NO_AUDIT
DEFSEG                       NO_AUDIT   DEFSYS                      NO_AUDIT
DELETE.A                     NO_AUDIT   DELETE.B                    NO_AUDIT
DESTAGE                      NO_AUDIT   DETACH.B                    NO_AUDIT
DETACH.G                     NO_AUDIT   DIAL                        NO_AUDIT
DISABLE.A                    NO_AUDIT   DISABLE.B                   NO_AUDIT
DISABLE.F                    NO_AUDIT   DISASSOCIATE                NO_AUDIT
DISCARD                      NO_AUDIT   DISCONNECT                  NO_AUDIT
DISPLAY.C                    NO_AUDIT   DISPLAY.E                   NO_AUDIT
DISPLAY.G                    NO_AUDIT   DRAIN.B                     NO_AUDIT
DRAIN.D                      NO_AUDIT   DUMP.C                      NO_AUDIT
DUMP.E                       NO_AUDIT   DUMP.G                      NO_AUDIT
DUPLEX                       NO_AUDIT   ECHO                        NO_AUDIT
ENABLE.A                     NO_AUDIT   ENABLE.B                    NO_AUDIT
ENABLE.F                     NO_AUDIT   EXPLORE                     NO_AUDIT
EXTERNAL                     NO_AUDIT   FLASHCOPY                   NO_AUDIT
FLUSH                        NO_AUDIT   FOR.C                       NO_AUDIT
FOR.G                        NO_AUDIT   FORCE                       NO_AUDIT
FORWARD                      NO_AUDIT   FREE.B                      NO_AUDIT
FREE.D                       NO_AUDIT   GIVE                        NO_AUDIT
HALT                         NO_AUDIT   HOLD.B                      NO_AUDIT
HOLD.D                       NO_AUDIT   HYPERSWAP                   NO_AUDIT
INDICATE.B                   NO_AUDIT   INDICATE.C                  NO_AUDIT
INDICATE.E                   NO_AUDIT   INDICATE.G                  NO_AUDIT
IPL                          NO_AUDIT   LINK                        NO_AUDIT
LOADBUF                      NO_AUDIT   LOADVFCB                    NO_AUDIT
LOCATE.C                     NO_AUDIT   LOCATE.E                    NO_AUDIT
LOCATEVM                     NO_AUDIT   LOCK                        NO_AUDIT
LOGON                        NO_AUDIT   LOGOFF                      NO_AUDIT
```

Figure 7. Sample Output from the SETEVENT LIST Command for z/VM (Part 1 of 7)

```
MESSAGE.A                 NO_AUDIT   MESSAGE.B                 NO_AUDIT
MESSAGE.ANY               NO_AUDIT   MODIFY.A                  NO_AUDIT
MODIFY.B                  NO_AUDIT   MONITOR.A                 NO_AUDIT
MONITOR.E                 NO_AUDIT   MSGNOH                    NO_AUDIT
NOTREADY                  NO_AUDIT   ORDER.D                   NO_AUDIT
ORDER.G                   NO_AUDIT   PURGE.A                   NO_AUDIT
PURGE.B                   NO_AUDIT   PURGE.C                   NO_AUDIT
PURGE.D                   NO_AUDIT   PURGE.E                   NO_AUDIT
PURGE.G                   NO_AUDIT   READY                     NO_AUDIT
RECORDING.A               NO_AUDIT   RECORDING.B               NO_AUDIT
RECORDING.C               NO_AUDIT   RECORDING.E               NO_AUDIT
RECORDING.F               NO_AUDIT   REDEFINE                  NO_AUDIT
REFRESH                   NO_AUDIT   RELSPACE                  NO_AUDIT
REPEAT                    NO_AUDIT   REQUEST                   NO_AUDIT
RESET.B                   NO_AUDIT   RESET.G                   NO_AUDIT
RESTART.A                 NO_AUDIT   RESTART.B                 NO_AUDIT
RESTART.G                 NO_AUDIT   RETAIN                    NO_AUDIT
REWIND                    NO_AUDIT   SAVESEG                   NO_AUDIT
SAVESYS                   NO_AUDIT   SCHEDULE                  NO_AUDIT
SCREEN                    NO_AUDIT   SEND.C                    NO_AUDIT
SEND.G                    NO_AUDIT   SHUTDOWN                  NO_AUDIT
SIGNAL.A                  NO_AUDIT   SIGNAL.C                  NO_AUDIT
SIGNAL.G                  NO_AUDIT   SILENTLY                  NO_AUDIT
SLEEP                     NO_AUDIT   SMSG                      NO_AUDIT
SNAPDUMP                  NO_AUDIT   SPACE                     NO_AUDIT
SPOOL                     NO_AUDIT   SPXTAPE.D                NO_AUDIT
SPXTAPE.E                 NO_AUDIT   SPXTAPE.G                NO_AUDIT
START.B                   NO_AUDIT   START.D                   NO_AUDIT
STOP                      NO_AUDIT   STORE.C                   NO_AUDIT
STORE.G                   NO_AUDIT   SYNCMDRS.A                NO_AUDIT
SYNCMDRS.B                NO_AUDIT   SYNCMDRS.F                NO_AUDIT
SYSTEM                    NO_AUDIT   TAG                       NO_AUDIT
TERMINAL                  NO_AUDIT   TRACE                     NO_AUDIT
TRANSFER.D                NO_AUDIT   TRANSFER.G                NO_AUDIT
TRSAVE.A                  NO_AUDIT   TRSAVE.C                  NO_AUDIT
TRSOURCE                  NO_AUDIT   UNCOUPLE                  NO_AUDIT
UNDIAL                    NO_AUDIT   UNLOCK                    NO_AUDIT
VARY                      NO_AUDIT   VDELETE                   NO_AUDIT
VINPUT                    NO_AUDIT   VMDUMP                    NO_AUDIT
VMRELOCATE                NO_AUDIT   WARNING.A                 NO_AUDIT
WARNING.B                 NO_AUDIT   WARNING.C                 NO_AUDIT
XAUTOLOG.A                NO_AUDIT   XAUTOLOG.B                NO_AUDIT
XAUTOLOG.G                NO_AUDIT   XLINK.A                   NO_AUDIT
XLINK.B                   NO_AUDIT   QUERY.ABEND               NO_AUDIT
```

Figure 8. Sample Output from the SETEVENT LIST Command for z/VM (Part 2 of 7)

```
QUERY.ACCOUNT               NO_AUDIT   QUERY.ADJUNCT               NO_AUDIT
QUERY.AGELIST.A             NO_AUDIT   QUERY.AGELIST.E             NO_AUDIT
QUERY.ALL                   NO_AUDIT   QUERY.ALLOC                 NO_AUDIT
QUERY.BYUSER.B              NO_AUDIT   QUERY.BYUSER.E              NO_AUDIT
QUERY.BYUSER.ANY            NO_AUDIT   QUERY.CACHE                 NO_AUDIT
QUERY.CACHEFW               NO_AUDIT   QUERY.CAPABILITY.A          NO_AUDIT
QUERY.CAPABILITY.B          NO_AUDIT   QUERY.CAPABILITY.C          NO_AUDIT
QUERY.CAPABILITY.E          NO_AUDIT   QUERY.CFLINKS.A             NO_AUDIT
QUERY.CFLINKS.B             NO_AUDIT   QUERY.CFLINKS.G             NO_AUDIT
QUERY.CHPID                 NO_AUDIT   QUERY.CHPIDS.B              NO_AUDIT
QUERY.CHPIDS.E              NO_AUDIT   QUERY.CHPIDV                NO_AUDIT
QUERY.CMDLIMIT.A            NO_AUDIT   QUERY.CMDLIMIT.B            NO_AUDIT
QUERY.COLLECT               NO_AUDIT   QUERY.COMMANDS              NO_AUDIT
QUERY.CONCOPY               NO_AUDIT   QUERY.CONFIGMODE.B          NO_AUDIT
QUERY.CONFIGMODE.E          NO_AUDIT   QUERY.CONTROLLER            NO_AUDIT
QUERY.CONV                  NO_AUDIT   QUERY.CPCHECKING.A          NO_AUDIT
QUERY.CPCHECKING.C          NO_AUDIT   QUERY.CPCHECKING.E          NO_AUDIT
QUERY.CPCMDS.A              NO_AUDIT   QUERY.CPCMDS.C              NO_AUDIT
QUERY.CPCMDS.E              NO_AUDIT   QUERY.CPDISKS               NO_AUDIT
QUERY.CPLANGUAGE            NO_AUDIT   QUERY.CPLANGLIST            NO_AUDIT
QUERY.CPLEVEL               NO_AUDIT   QUERY.CPLOAD.A              NO_AUDIT
QUERY.CPLOAD.B              NO_AUDIT   QUERY.CPLOAD.E              NO_AUDIT
QUERY.CPOWNED               NO_AUDIT   QUERY.CPPROTECT             NO_AUDIT
QUERY.CPSERVICE.C           NO_AUDIT   QUERY.CPSERVICE.E           NO_AUDIT
QUERY.CPTRACE.A             NO_AUDIT   QUERY.CPTRACE.C             NO_AUDIT
QUERY.CPTRACE.E             NO_AUDIT   QUERY.CPUAFFINITY           NO_AUDIT
QUERY.CPUID                 NO_AUDIT   QUERY.CPXLOAD.A             NO_AUDIT
QUERY.CPXLOAD.C             NO_AUDIT   QUERY.CPXLOAD.E             NO_AUDIT
QUERY.CRYPTO.A              NO_AUDIT   QUERY.CRYPTO.B              NO_AUDIT
QUERY.CRYPTO.C              NO_AUDIT   QUERY.CRYPTO.E              NO_AUDIT
QUERY.CTCA                  NO_AUDIT   QUERY.CU                    NO_AUDIT
QUERY.DASD                  NO_AUDIT   QUERY.DASDFW                NO_AUDIT
QUERY.DATEFORMAT            NO_AUDIT   QUERY.DIAGNOSE.A            NO_AUDIT
QUERY.DIAGNOSE.C            NO_AUDIT   QUERY.DIAGNOSE.E            NO_AUDIT
QUERY.DUPLEX                NO_AUDIT   QUERY.DUMP                  NO_AUDIT
QUERY.DUMPDEV               NO_AUDIT   QUERY.DYNAMIC_IO.B          NO_AUDIT
QUERY.DYNAMIC_IO.E          NO_AUDIT   QUERY.D8ONECMD.A            NO_AUDIT
QUERY.D8ONECMD.C            NO_AUDIT   QUERY.D8ONECMD.E            NO_AUDIT
QUERY.D8ONECMD.G            NO_AUDIT   QUERY.EDEVICE               NO_AUDIT
QUERY.ENCRYPT.A             NO_AUDIT   QUERY.ENCRYPT.C             NO_AUDIT
QUERY.ENCRYPT.E             NO_AUDIT   QUERY.EQID                  NO_AUDIT
QUERY.EXITS.A               NO_AUDIT   QUERY.EXITS.C               NO_AUDIT
QUERY.EXITS.E               NO_AUDIT   QUERY.FCP                   NO_AUDIT
QUERY.FENCES                NO_AUDIT   QUERY.FILES.D               NO_AUDIT
QUERY.FILES.G               NO_AUDIT   QUERY.FLASHCOPY             NO_AUDIT
QUERY.FRAMES.A              NO_AUDIT   QUERY.FRAMES.B              NO_AUDIT
QUERY.FRAMES.E              NO_AUDIT   QUERY.GATEWAY               NO_AUDIT
QUERY.GRAF                  NO_AUDIT   QUERY.HCD                   NO_AUDIT
QUERY.HOLD.B                NO_AUDIT   QUERY.HOLD.D                NO_AUDIT
QUERY.HOTIO                 NO_AUDIT   QUERY.HYPERSWAP             NO_AUDIT
QUERY.ICLNAME.A             NO_AUDIT   QUERY.ICLNAME.C             NO_AUDIT
QUERY.ICLNAME.E             NO_AUDIT   QUERY.IMG.A                 NO_AUDIT
QUERY.IMG.B                 NO_AUDIT   QUERY.IMG.C                 NO_AUDIT
QUERY.IMG.D                 NO_AUDIT   QUERY.IMG.E                 NO_AUDIT
QUERY.IOASSIST.B            NO_AUDIT   QUERY.IOASSIST.G            NO_AUDIT
QUERY.IO_OPT.B              NO_AUDIT   QUERY.IO_OPT.G              NO_AUDIT
QUERY.IOPRIORITY.A          NO_AUDIT   QUERY.IOPRIORITY.E          NO_AUDIT
QUERY.IPLPARMS              NO_AUDIT   QUERY.ISFC                  NO_AUDIT
QUERY.ISLINK                NO_AUDIT   QUERY.IUCV.B               NO_AUDIT
QUERY.IUCV.G                NO_AUDIT   QUERY.JOURNAL.A             NO_AUDIT
QUERY.JOURNAL.E             NO_AUDIT   QUERY.KEYALIAS              NO_AUDIT
QUERY.LDEVS.B               NO_AUDIT   QUERY.LDEVS.G               NO_AUDIT
QUERY.LGRWAIT               NO_AUDIT   QUERY.LINES                 NO_AUDIT
QUERY.LINKS                 NO_AUDIT   QUERY.LKFAC                 NO_AUDIT
QUERY.LKFACR                NO_AUDIT   QUERY.LOADDEV               NO_AUDIT
QUERY.LOGMSG.A              NO_AUDIT   QUERY.LOGMSG.B              NO_AUDIT
QUERY.LOGMSG.C              NO_AUDIT   QUERY.LOGMSG.D              NO_AUDIT
QUERY.LOGMSG.E              NO_AUDIT   QUERY.LOGMSG.F              NO_AUDIT
QUERY.LOGMSG.G              NO_AUDIT   QUERY.LAN.B                 NO_AUDIT
QUERY.LAN.G                 NO_AUDIT   QUERY.LPARS                 NO_AUDIT
QUERY.LSYSTEM               NO_AUDIT   QUERY.MAXLDEV               NO_AUDIT
```

Figure 9. Sample Output from the SETEVENT LIST Command for z/VM (Part 3 of 7)

```
QUERY.MAXSPOOL.D            NO_AUDIT   QUERY.MAXSPOOL.G            NO_AUDIT
QUERY.MAXUSERS              NO_AUDIT   QUERY.MDCACHE.B             NO_AUDIT
QUERY.MDCACHE.G             NO_AUDIT   QUERY.MDISK                 NO_AUDIT
QUERY.MEMASSIST.B           NO_AUDIT   QUERY.MEMASSIST.G           NO_AUDIT
QUERY.MITIME.A              NO_AUDIT   QUERY.MITIME.B              NO_AUDIT
QUERY.MONDATA               NO_AUDIT   QUERY.MONITOR.A             NO_AUDIT
QUERY.MONITOR.E             NO_AUDIT   QUERY.MSS                   NO_AUDIT
QUERY.MULTITHREAD.A         NO_AUDIT   QUERY.MULTITHREAD.B         NO_AUDIT
QUERY.MULTITHREAD.C         NO_AUDIT   QUERY.MULTITHREAD.E         NO_AUDIT
QUERY.MULTITHREAD.G         NO_AUDIT   QUERY.NAMES.A               NO_AUDIT
QUERY.NAMES.B               NO_AUDIT   QUERY.NAMES.C               NO_AUDIT
QUERY.NAMES.D               NO_AUDIT   QUERY.NAMES.E               NO_AUDIT
QUERY.NAMES.F               NO_AUDIT   QUERY.NAMES.G               NO_AUDIT
QUERY.NEW_DEVICES           NO_AUDIT   QUERY.NIC                   NO_AUDIT
QUERY.NLS                   NO_AUDIT   QUERY.NSS                   NO_AUDIT
QUERY.NVS                   NO_AUDIT   QUERY.OBSERVER.A            NO_AUDIT
QUERY.OBSERVER.B            NO_AUDIT   QUERY.OBSERVER.C            NO_AUDIT
QUERY.OBSERVER.G            NO_AUDIT   QUERY.OSA                   NO_AUDIT
QUERY.PCIFUNCTION           NO_AUDIT   QUERY.PAGING.B              NO_AUDIT
QUERY.PAGING.E              NO_AUDIT   QUERY.PASSWORD              NO_AUDIT
QUERY.PATHS.B               NO_AUDIT   QUERY.PATHS.E               NO_AUDIT
QUERY.PAV                   NO_AUDIT   QUERY.PENDING               NO_AUDIT
QUERY.PINNED                NO_AUDIT   QUERY.PF                    NO_AUDIT
QUERY.PORT                  NO_AUDIT   QUERY.PRINTER.D             NO_AUDIT
QUERY.PRINTER.G             NO_AUDIT   QUERY.PRIVCLASS.C           NO_AUDIT
QUERY.PRIVCLASS.E           NO_AUDIT   QUERY.PRIVCLASS.ANY         NO_AUDIT
QUERY.PROCESSORS.A          NO_AUDIT   QUERY.PROCESSORS.B          NO_AUDIT
QUERY.PROCESSORS.C          NO_AUDIT   QUERY.PROCESSORS.E          NO_AUDIT
QUERY.PRODUCT.C             NO_AUDIT   QUERY.PRODUCT.E             NO_AUDIT
QUERY.PROMPT                NO_AUDIT   QUERY.PSWTRANS              NO_AUDIT
QUERY.PUNCH.D               NO_AUDIT   QUERY.PUNCH.G               NO_AUDIT
QUERY.PVMSG                 NO_AUDIT   QUERY.QIOASSIST.B           NO_AUDIT
QUERY.QIOASSIST.G           NO_AUDIT   QUERY.QUICKDSP.A            NO_AUDIT
QUERY.QUICKDSP.E            NO_AUDIT   QUERY.READER.D              NO_AUDIT
QUERY.READER.G              NO_AUDIT   QUERY.RECORDING.A           NO_AUDIT
QUERY.RECORDING.B           NO_AUDIT   QUERY.RECORDING.C           NO_AUDIT
QUERY.RECORDING.E           NO_AUDIT   QUERY.RECORDING.F           NO_AUDIT
QUERY.REORDER.B             NO_AUDIT   QUERY.REORDER.E             NO_AUDIT
QUERY.RELODOMAIN.A          NO_AUDIT   QUERY.RELODOMAIN.B          NO_AUDIT
QUERY.RELODOMAIN.C          NO_AUDIT   QUERY.RELODOMAIN.E          NO_AUDIT
QUERY.RESERVED.A            NO_AUDIT   QUERY.RESERVED.E            NO_AUDIT
QUERY.RESOURCE              NO_AUDIT   QUERY.RESPOOL.A             NO_AUDIT
QUERY.RESPOOL.E             NO_AUDIT   QUERY.RETRIEVE              NO_AUDIT
QUERY.RSAW                  NO_AUDIT   QUERY.SCMBKS.B              NO_AUDIT
QUERY.SCMBKS.E              NO_AUDIT   QUERY.SCMEASURE.B           NO_AUDIT
QUERY.SCMEASURE.E           NO_AUDIT   QUERY.SCREEN                NO_AUDIT
QUERY.SDF.A                 NO_AUDIT   QUERY.SDF.B                 NO_AUDIT
QUERY.SDF.C                 NO_AUDIT   QUERY.SDF.D                 NO_AUDIT
QUERY.SDF.E                 NO_AUDIT   QUERY.SDF.G                 NO_AUDIT
QUERY.SECUSER.A             NO_AUDIT   QUERY.SECUSER.B             NO_AUDIT
QUERY.SECUSER.C             NO_AUDIT   QUERY.SECUSER.G             NO_AUDIT
QUERY.SET                   NO_AUDIT   QUERY.SHARE.A               NO_AUDIT
QUERY.SHARE.E               NO_AUDIT   QUERY.SHUTDOWN.A            NO_AUDIT
QUERY.SHUTDOWN.C            NO_AUDIT   QUERY.SHUTDOWN.G            NO_AUDIT
QUERY.SHUTDOWNTIME.A        NO_AUDIT   QUERY.SHUTDOWNTIME.C        NO_AUDIT
QUERY.SIGNAL                NO_AUDIT   QUERY.SIGNALS               NO_AUDIT
QUERY.SPACES.E              NO_AUDIT   QUERY.SPACES.G              NO_AUDIT
QUERY.SRM.A                 NO_AUDIT   QUERY.SRM.E                 NO_AUDIT
QUERY.SSI.B                 NO_AUDIT   QUERY.SSI.E                 NO_AUDIT
QUERY.STGEXEMPT.A           NO_AUDIT   QUERY.STGEXEMPT.B           NO_AUDIT
QUERY.STGEXEMPT.C           NO_AUDIT   QUERY.STGEXEMPT.E           NO_AUDIT
QUERY.STGEXEMPT.G           NO_AUDIT   QUERY.STGLIMIT.A            NO_AUDIT
QUERY.STGLIMIT.B            NO_AUDIT   QUERY.STGLIMIT.C            NO_AUDIT
QUERY.STGLIMIT.E            NO_AUDIT   QUERY.STORAGE.A             NO_AUDIT
QUERY.STORAGE.B             NO_AUDIT   QUERY.STORAGE.E             NO_AUDIT
QUERY.STP                   NO_AUDIT   QUERY.SUBSTITUTE            NO_AUDIT
QUERY.SWITCHES              NO_AUDIT   QUERY.SXSPAGES.A            NO_AUDIT
QUERY.SXSPAGES.B            NO_AUDIT   QUERY.SXSPAGES.E            NO_AUDIT
QUERY.SXSSTORAGE.A          NO_AUDIT   QUERY.SXSSTORAGE.B          NO_AUDIT
QUERY.SXSSTORAGE.E          NO_AUDIT   QUERY.SYSASCII              NO_AUDIT
QUERY.SYSCONTROL.A          NO_AUDIT   QUERY.SYSCONTROL.E          NO_AUDIT
QUERY.SYSOPER               NO_AUDIT   QUERY.SYSTEM                NO_AUDIT
```

Figure 10. Sample Output from the SETEVENT LIST Command for z/VM (Part 4 of 7)

```
QUERY.TAG                   NO_AUDIT   QUERY.TAPES                 NO_AUDIT
QUERY.TDISK                 NO_AUDIT   QUERY.TDISKCLR              NO_AUDIT
QUERY.TERMINAL              NO_AUDIT   QUERY.THROTTLE.B            NO_AUDIT
QUERY.THROTTLE.E            NO_AUDIT   QUERY.TIME                  NO_AUDIT
QUERY.TIMEZONES             NO_AUDIT   QUERY.TOKEN                 NO_AUDIT
QUERY.TRACE                 NO_AUDIT   QUERY.TRACEFRAMES.A         NO_AUDIT
QUERY.TRACEFRAMES.B         NO_AUDIT   QUERY.TRACEFRAMES.C         NO_AUDIT
QUERY.TRACEFRAMES.E         NO_AUDIT   QUERY.TRFILES.A             NO_AUDIT
QUERY.TRFILES.C             NO_AUDIT   QUERY.TRFILES.D             NO_AUDIT
QUERY.TRFILES.E             NO_AUDIT   QUERY.TRFILES.G             NO_AUDIT
QUERY.TRSAVE.A              NO_AUDIT   QUERY.TRSAVE.C              NO_AUDIT
QUERY.TRSAVE.E              NO_AUDIT   QUERY.TRSAVE.G              NO_AUDIT
QUERY.TRSOURCE.A            NO_AUDIT   QUERY.TRSOURCE.C            NO_AUDIT
QUERY.TRSOURCE.E            NO_AUDIT   QUERY.TRSOURCE.G            NO_AUDIT
QUERY.UCR.A                 NO_AUDIT   QUERY.UCR.B                 NO_AUDIT
QUERY.UCR.C                 NO_AUDIT   QUERY.UNDERSCORE            NO_AUDIT
QUERY.UNRESOLVED.A          NO_AUDIT   QUERY.UNRESOLVED.C          NO_AUDIT
QUERY.UNRESOLVED.E          NO_AUDIT   QUERY.UR                    NO_AUDIT
QUERY.USERID                NO_AUDIT   QUERY.USERS                 NO_AUDIT
QUERY.VARIABLE              NO_AUDIT   QUERY.VCONFIG               NO_AUDIT
QUERY.VDISK                 NO_AUDIT   QUERY.VMDUMP                NO_AUDIT
QUERY.VMLAN                 NO_AUDIT   QUERY.VMRELOCATE.A          NO_AUDIT
QUERY.VMRELOCATE.B          NO_AUDIT   QUERY.VMRELOCATE.C          NO_AUDIT
QUERY.VMRELOCATE.E          NO_AUDIT   QUERY.VMSG                  NO_AUDIT
QUERY.VSWITCH.B             NO_AUDIT   QUERY.VSWITCH.G             NO_AUDIT
QUERY.VTOD.A                NO_AUDIT   QUERY.VTOD.B                NO_AUDIT
QUERY.VTOD.G                NO_AUDIT   QUERY.WRKALLEG              NO_AUDIT
QUERY.XSTORAGE.B            NO_AUDIT   QUERY.XSTORAGE.E            NO_AUDIT
QUERY.V.ALL                 NO_AUDIT   QUERY.V.CHPID               NO_AUDIT
QUERY.V.CONSOLE             NO_AUDIT   QUERY.V.CPUS                NO_AUDIT
QUERY.V.CRYPTO              NO_AUDIT   QUERY.V.CTCA                NO_AUDIT
QUERY.V.DASD                NO_AUDIT   QUERY.V.DUPLEX              NO_AUDIT
QUERY.V.FCP                 NO_AUDIT   QUERY.V.FLASHCOPY           NO_AUDIT
QUERY.V.GRAF                NO_AUDIT   QUERY.V.LINES               NO_AUDIT
QUERY.V.MSGDEVICES          NO_AUDIT   QUERY.V.MSGPROC             NO_AUDIT
QUERY.V.NIC                 NO_AUDIT   QUERY.V.OSA                 NO_AUDIT
QUERY.V.PAV                 NO_AUDIT   QUERY.V.PCIFUNCTION         NO_AUDIT
QUERY.V.PRINTER             NO_AUDIT   QUERY.V.PUNCH               NO_AUDIT
QUERY.V.READER              NO_AUDIT   QUERY.V.STORAGE             NO_AUDIT
QUERY.V.SWITCHES            NO_AUDIT   QUERY.V.SYSASCII            NO_AUDIT
QUERY.V.TAPES               NO_AUDIT   QUERY.V.UR                  NO_AUDIT
QUERY.V.XSTORAGE            NO_AUDIT   QUERY.VIRTUAL.B             NO_AUDIT
QUERY.VIRTUAL.G             NO_AUDIT   SET.ABEND                   NO_AUDIT
SET.ACCOUNT                 NO_AUDIT   SET.ADJUNCTS                NO_AUDIT
SET.AGELIST                 NO_AUDIT   SET.AUTOPOLL                NO_AUDIT
SET.CACHE                   NO_AUDIT   SET.CACHEFW                 NO_AUDIT
SET.CFLINK.A                NO_AUDIT   SET.CFLINK.B                NO_AUDIT
SET.CFLINK.G                NO_AUDIT   SET.CMDLIMIT                NO_AUDIT
SET.CONCEAL                 NO_AUDIT   SET.CONFIGMODE              NO_AUDIT
SET.CPCHECKING.A            NO_AUDIT   SET.CPCHECKING.C            NO_AUDIT
SET.CPCONIO                 NO_AUDIT   SET.CPLANGUAGE.B            NO_AUDIT
SET.CPLANGUAGE.G            NO_AUDIT   SET.CPPROTECT               NO_AUDIT
SET.CPTRACE.A               NO_AUDIT   SET.CPTRACE.C               NO_AUDIT
SET.CPUAFFINITY             NO_AUDIT   SET.CU                      NO_AUDIT
SET.CPUID                   NO_AUDIT   SET.DASDFW                  NO_AUDIT
SET.DASD                    NO_AUDIT   SET.DATEFORMAT.B            NO_AUDIT
SET.DATEFORMAT.G            NO_AUDIT   SET.DEVICES                 NO_AUDIT
SET.DIALDROP                NO_AUDIT   SET.DUMP                    NO_AUDIT
SET.DUMPDEV                 NO_AUDIT   SET.DYNAMIC_IO              NO_AUDIT
SET.D8ONECMD.A              NO_AUDIT   SET.D8ONECMD.G              NO_AUDIT
SET.ECMODE                  NO_AUDIT   SET.EDEVICE                 NO_AUDIT
SET.EMSG                    NO_AUDIT   SET.ENCRYPT                 NO_AUDIT
SET.HOTIO                   NO_AUDIT   SET.IMSG                    NO_AUDIT
```

Figure 11. Sample Output from the SETEVENT LIST Command for z/VM (Part 5 of 7)

```
SET.IOASSIST.B              NO_AUDIT   SET.IOASSIST.G              NO_AUDIT
SET.IOCDS_ACTIVE            NO_AUDIT   SET.IO_OPT.B                NO_AUDIT
SET.IO_OPT.G                NO_AUDIT   SET.IOPRIORITY              NO_AUDIT
SET.IPLPARMS                NO_AUDIT   SET.JOURNAL                 NO_AUDIT
SET.KEYALIAS                NO_AUDIT   SET.LAN.B                   NO_AUDIT
SET.LAN.G                   NO_AUDIT   SET.LGRWAIT                 NO_AUDIT
SET.LINEDIT                 NO_AUDIT   SET.LKFAC                   NO_AUDIT
SET.LKFACR                  NO_AUDIT   SET.LOADDEV                 NO_AUDIT
SET.LOGMSG                  NO_AUDIT   SET.LSYSTEM                 NO_AUDIT
SET.MACHINE                 NO_AUDIT   SET.MAXLDEV                 NO_AUDIT
SET.MAXUSERS                NO_AUDIT   SET.MDCACHE.B               NO_AUDIT
SET.MDCACHE.G               NO_AUDIT   SET.MEMASSIST.B             NO_AUDIT
SET.MEMASSIST.G             NO_AUDIT   SET.MSG                     NO_AUDIT
SET.MSGFACIL                NO_AUDIT   SET.MIH                     NO_AUDIT
SET.MITIME.A                NO_AUDIT   SET.MITIME.B                NO_AUDIT
SET.MODE.A                  NO_AUDIT   SET.MODE.F                  NO_AUDIT
SET.MONDATA                 NO_AUDIT   SET.MULTITHREAD             NO_AUDIT
SET.NEW_DEVICES             NO_AUDIT   SET.NIC.B                   NO_AUDIT
SET.NIC.G                   NO_AUDIT   SET.NOPDATA                 NO_AUDIT
SET.NVS                     NO_AUDIT   SET.OBSERVER.A              NO_AUDIT
SET.OBSERVER.C              NO_AUDIT   SET.OBSERVER.G              NO_AUDIT
SET.PAGEX                   NO_AUDIT   SET.PAGING                  NO_AUDIT
SET.PASSWORD                NO_AUDIT   SET.PF                      NO_AUDIT
SET.PORT                    NO_AUDIT   SET.PRIVCLASS.C             NO_AUDIT
SET.PRIVCLASS.ANY           NO_AUDIT   SET.PRODUCT.C               NO_AUDIT
SET.PRODUCT.E               NO_AUDIT   SET.PROMPT                  NO_AUDIT
SET.PSWTRANS                NO_AUDIT   SET.QIOASSIST.B             NO_AUDIT
SET.QIOASSIST.G             NO_AUDIT   SET.QUICKDSP                NO_AUDIT
SET.RECORD                  NO_AUDIT   SET.RDEVICE                 NO_AUDIT
SET.REORDER                 NO_AUDIT   SET.RESERVED                NO_AUDIT
SET.RESPOOL                 NO_AUDIT   SET.RETRIEVE.C              NO_AUDIT
SET.RETRIEVE.E              NO_AUDIT   SET.RETRIEVE.G              NO_AUDIT
SET.RUN                     NO_AUDIT   SET.SCMEASURE.B             NO_AUDIT
SET.SCMEASURE.E             NO_AUDIT   SET.SECUSER.A               NO_AUDIT
SET.SECUSER.C               NO_AUDIT   SET.SECUSER.G               NO_AUDIT
SET.SHARE                   NO_AUDIT   SET.SHARED                  NO_AUDIT
SET.SHUTSIGNAL              NO_AUDIT   SET.SHUTDOWNTIME.A          NO_AUDIT
SET.SHUTDOWNTIME.C          NO_AUDIT   SET.SIGNAL.A                NO_AUDIT
SET.SIGNAL.C                NO_AUDIT   SET.SMSG                    NO_AUDIT
SET.SRM                     NO_AUDIT   SET.SSI                     NO_AUDIT
SET.STGEXEMPT.A             NO_AUDIT   SET.STGEXEMPT.B             NO_AUDIT
SET.STGEXEMPT.C             NO_AUDIT   SET.STGLIMIT.A              NO_AUDIT
SET.STGLIMIT.B              NO_AUDIT   SET.STGLIMIT.C              NO_AUDIT
SET.STORAGE                 NO_AUDIT   SET.SVC76                   NO_AUDIT
SET.SYSCONTROL              NO_AUDIT   SET.SYSOPER                 NO_AUDIT
SET.TAPE                    NO_AUDIT   SET.THROTTLE                NO_AUDIT
SET.TIMEBOMB                NO_AUDIT   SET.TIMER                   NO_AUDIT
SET.TIMEZONE                NO_AUDIT   SET.TOKEN.B                 NO_AUDIT
SET.TOKEN.E                 NO_AUDIT   SET.TRACEFRAMES             NO_AUDIT
SET.UNDERSCORE              NO_AUDIT   SET.VARIABLE                NO_AUDIT
SET.VCONFIG                 NO_AUDIT   SET.VDISK                   NO_AUDIT
SET.VMCONIO                 NO_AUDIT   SET.VMLAN                   NO_AUDIT
SET.VMRELOCATE              NO_AUDIT   SET.VSWITCH                 NO_AUDIT
SET.VTOD.A                  NO_AUDIT   SET.VTOD.B                  NO_AUDIT
SET.VTOD.G                  NO_AUDIT   SET.WNG                     NO_AUDIT
SET.WRKALLEG                NO_AUDIT   SET.370ACCOM                NO_AUDIT
```

Figure 12. Sample Output from the SETEVENT LIST Command for z/VM (Part 6 of 7)

```
DIAG000                     NO_AUDIT   DIAG004                     NO_AUDIT
DIAG008                     NO_AUDIT   DIAG00C                     NO_AUDIT
DIAG010                     NO_AUDIT   DIAG014                     NO_AUDIT
DIAG018                     NO_AUDIT   DIAG020                     NO_AUDIT
DIAG024                     NO_AUDIT   DIAG028                     NO_AUDIT
DIAG034                     NO_AUDIT   DIAG03C                     NO_AUDIT
DIAG040                     NO_AUDIT   DIAG044                     NO_AUDIT
DIAG048                     NO_AUDIT   DIAG04C                     NO_AUDIT
DIAG054                     NO_AUDIT   DIAG058                     NO_AUDIT
DIAG05C                     NO_AUDIT   DIAG060                     NO_AUDIT
DIAG064                     NO_AUDIT   DIAG068                     NO_AUDIT
DIAG070                     NO_AUDIT   DIAG074                     NO_AUDIT
DIAG07C                     NO_AUDIT   DIAG084                     NO_AUDIT
DIAG088                     NO_AUDIT   DIAG08C                     NO_AUDIT
DIAG090                     NO_AUDIT   DIAG094                     NO_AUDIT
DIAG098                     NO_AUDIT   DIAG09C                     NO_AUDIT
DIAG0A0                     NO_AUDIT   DIAG0A4                     NO_AUDIT
DIAG0A8                     NO_AUDIT   DIAG0B0                     NO_AUDIT
DIAG0B4                     NO_AUDIT   DIAG0B8                     NO_AUDIT
DIAG0BC                     NO_AUDIT   DIAG0C8                     NO_AUDIT
DIAG0CC                     NO_AUDIT   DIAG0D0                     NO_AUDIT
DIAG0D4                     NO_AUDIT   DIAG0D8                     NO_AUDIT
DIAG0DC                     NO_AUDIT   DIAG0E0                     NO_AUDIT
DIAG0E4                     NO_AUDIT   DIAG0EC                     NO_AUDIT
DIAG0F8                     NO_AUDIT   DIAG204                     NO_AUDIT
DIAG210                     NO_AUDIT   DIAG214                     NO_AUDIT
DIAG218                     NO_AUDIT   DIAG220                     NO_AUDIT
DIAG224                     NO_AUDIT   DIAG238                     NO_AUDIT
DIAG23C                     NO_AUDIT   DIAG240                     NO_AUDIT
DIAG244                     NO_AUDIT   DIAG248                     NO_AUDIT
DIAG250                     NO_AUDIT   DIAG254                     NO_AUDIT
DIAG258                     NO_AUDIT   DIAG25C                     NO_AUDIT
DIAG260                     NO_AUDIT   DIAG264                     NO_AUDIT
DIAG268                     NO_AUDIT   DIAG26C                     NO_AUDIT
DIAG270                     NO_AUDIT   DIAG274                     NO_AUDIT
DIAG278                     NO_AUDIT   DIAG27C                     NO_AUDIT
DIAG280                     NO_AUDIT   DIAG288                     NO_AUDIT
DIAG290                     NO_AUDIT   DIAG29C                     NO_AUDIT
DIAG2A0                     NO_AUDIT   DIAG2A4                     NO_AUDIT
DIAG2A8                     NO_AUDIT   DIAG2AC                     NO_AUDIT
DIAG2C0                     NO_AUDIT   DIAG2C4                     NO_AUDIT
DIAG2CC                     NO_AUDIT   DIAG2E0                     NO_AUDIT
DIAG2FC                     NO_AUDIT   DIAG308                     NO_AUDIT
DIAG318                     NO_AUDIT   IUCVCON                     NO_AUDIT
IUCVSEV                     NO_AUDIT   APPCCON                     NO_AUDIT
APPCPWVL                    NO_AUDIT   APPCSEV                     NO_AUDIT
SPF_CREATE                  NO_AUDIT   SPF_DELETE                  NO_AUDIT
SPF_OPEN                    NO_AUDIT   SDF_CREATE                  NO_AUDIT
SDF_DELETE                  NO_AUDIT   SDF_OPEN                    NO_AUDIT
UTLPRINT                    NO_AUDIT   MDISK                       NO_AUDIT
MAINTCCW                    NO_AUDIT   RSTDSEG                     NO_AUDIT
SNIFFER_MODE                NO_AUDIT   DIRECTRY_CMD                NO_AUDIT
SCIF                        NO_AUDIT   RDEVCTRL                    NO_AUDIT
STHYI-UTIL                  NO_AUDIT   STHYI-GUEST                 NO_AUDIT
STHYI-RESPOO                NO_AUDIT   DEFINE.MDISK                NO_AUDIT
RPISET126I SETEVENT COMPLETED SUCCESSFULLY.
```

Figure 13. Sample Output from the SETEVENT LIST Command for z/VM (Part 7 of 7)

Preventing Use of DIAL, UNDIAL, and MESSAGE Commands Before Logon

When a user issues a command before logging on, no user ID is associated with the command. This can make it difficult to determine who the user is. You can prevent users from issuing the DIAL, UNDIAL, and MESSAGE commands before logging on. To do this, issue the SETEVENT command with the NODIAL and NOPRELOGMSG operands, as follows:

```
SETEVENT  NODIAL  NOPRELOGMSG
```

Chapter 9. Protecting the Use of CP Commands, Diagnose Codes, and Other Functions

You can use RACF to protect certain CP commands, diagnose codes, and other functions with profile names in the VMCMD class, as described in Table 18 on page 127.

Some of these events are controllable through the use of a VMXEVENT profile. If you have activated a VMXEVENT profile that turns off control for a particular event, RACF will not be called—even if a profile exists in the VMCMD class.

Security Label Considerations for Profiles in the VMCMD Class

If a security label exists in the VMCMD profile protecting the resource, and the SECLABEL class is active, RACF compares the security label of the user with the security label contained in the profile, and grants access if the security label of the user is equal to or higher than the security label contained in the profile.

SYSSEC Considerations for VMCMD Requests

The SYSSEC macro, coded in the RACF module HCPRWA, can affect the final outcome of resource requests in the VMCMD class for the STORE.C, TRSOURCE, DEFINE.MDISK, DIAG0E4, XAUTOLOG.G, FOR, DIAGNOSE X'88', and DIAGNOSE X'A0' events. The SYSSEC macro defines the relationship between RACF's response to a resource access request and the final disposition of that request by z/VM. This relationship changes as you change the SYSSEC statement in HCPRWA.

For example, when a user issues a VMCMD command, there may be no RACF profile to protect that command. SYSSEC can be coded so that for this case, RACF will do one of the following:

- Allow access
- Disallow access
- Defer the access decision to z/VM

For DIAGNOSE X'88' and the FOR command, the SURROGAT check could fail in the RACF service machine. SYSSEC can be coded so that in this case RACF will defer the access decision to CP directory checking. The SURROGAT result is controlled by the "default" category of settings in the SYSSEC macro.

You should know how SYSSEC is affecting your VMCMD requests. See *z/VM: RACF Security Server Macros and Interfaces* for more information.

Protecting the STORE.C, TRSOURCE, and DIAG0E4 Events

The z/VM event names STORE.C, TRSOURCE, and DIAG0E4 refer to the STORE HOST command, TRSOURCE command, and DIAGNOSE code X'E4', respectively. z/VM has restrictions on the use of these events (for example, STORE HOST is limited to privilege class C users only), but you can further restrict access to these events with a VMCMD profile for each event.

1. Define the command as a profile in the VMCMD class and specify a UACC of NONE:

   ```
   RDEFINE  VMCMD  event_name  UACC(NONE)
   ```

2. Allow the appropriate users or groups to use the command by giving them READ access:

   ```
   PERMIT event_name CLASS(VMCMD) ID(user or group) ACCESS(READ)
   ```

   Note that if you do not give users an access authority of READ when creating the profile's access list, the users will not be able to use the command.

3. If the VMCMD class is not already active, use the SETROPTS command to activate it:

```
SETROPTS CLASSACT(VMCMD)
```

4. Make sure protection for the respective command name is active. (See Chapter 8, “Protecting z/VM with VMXEVENT Profiles,” on page 127.) It is on by default if an installation has not changed the setting in the currently active VMXEVENT profile by issuing, for example:

   ```
   RDEFINE VMXEVENT profile-name ADDMEM(command_name/NOCTL)
   ```

   If protection of the command name is not currently active, activate it by issuing:

   ```
   RALTER VMXEVENT profile-name DELMEM(event_name/NOCTL)
   SETEVENT REFRESH profile-name
   ```

Protecting the DEFINE.MDISK Command

The z/VM event name DEFINE.MDISK refers to the DEFINE.MDISK command. z/VM has restrictions on the use of the DEFINE.A event (for example, the DEFINE command may be limited to privilege class A users only), but you can further restrict access to this event by a VMCMD profile. To do so, follow these steps:

1. Define the command as a profile in the VMCMD class and specify a UACC of NONE:

   ```
   RDEFINE VMCMD DEFINE.MDISK UACC(NONE) AUDIT(ALL(READ))
   ```

2. Allow the appropriate users or groups to use the command by giving them UPDATE access:

   ```
   PERMIT DEFINE.MDISK ACCESS(UPDATE) CLASS(VMCMD) ID(user or group)
   ```

   Note: If you do not give users an access authority of UPDATE when creating the profile's access list, the users will not be able to use the command.

3. If the VMCMD class is not already active, use the SETROPTS command to activate it:

   ```
   SETROPTS CLASSACT(VMCMD)
   ```

Protecting the XAUTOLOG.G Command

For virtual machines that are logged on with the XAUTOLOG command, you can control which other privilege class G virtual machines can do the XAUTOLOG. To do this, create profiles named XAUTOLOG.*userid*, where *userid* is the virtual machine being logged on with the XAUTOLOG command. Give READ access to the privilege class G user ID that will be issuing the XAUTOLOG command.

For example, for USER1 to issue the command XAUTOLOG USER2, USER1 must have at least READ access authority to the VMCMD profile XAUTOLOG.USER2. No password is required to be entered.

1. Define the XAUTOLOG user ID as a profile in the VMCMD class and specify a UACC of NONE:

   ```
   RDEFINE  VMCMD  XAUTOLOG.userid  UACC(NONE)
   ```

2. Allow the appropriate users or groups to use the command by giving them READ access:

   ```
   PERMIT XAUTOLOG.userid CLASS(VMCMD) ID(user or group) ACCESS(READ)
   ```

   Note that if you do not give users an access authority of READ when creating the profile's access list, the users will not be able to use the command.

3. If the VMCMD class is not already active, use the SETROPTS command to activate it:

   ```
   SETROPTS CLASSACT(VMCMD)
   ```

For information on the XAUTOLOG command when the SECLABEL class is active, see “Security Label Considerations for Profiles in the VMCMD Class” on page 143 and “LOGON, AUTOLOG, and XAUTOLOG Commands” on page 262.

Protecting XAUTOLOG ON

Use of XAUTOLOG ON could be considered a security risk due to its capacity to give virtual machine access to a target device. You can control which virtual machines with class A or B authority can issue XAUTOLOG with the ON operand. To do so, create profiles named XAUTOLOG.ON.*target_userid.systemid*, where *target_userid* is the virtual machine that is being logged on with the XAUTOLOG command and *systemid* is the node onto which it is being logged. READ access is sufficient to grant that user ID the authority to log the specified *target_userid* on to the specified system at the provided rdev or vdev.

For example, for USER1 to issue the command XAUTOLOG USER2 ON 0020 on SYSTEM2, USER1 must have at least READ access authority to the VMCMD profile XAUTOLOG.ON.USER2.SYSTEM2.

1. Define the XAUTOLOG target user ID as a profile in the VMCMD class and specify a UACC of NONE:

   ```
   RDEFINE VMCMD XAUTOLOG.ON.target_userid.systemid UACC(NONE)
   ```

2. Allow the appropriate users or groups to use the command by giving them READ access:

   ```
   PERMIT XAUTOLOG.ON.target_userid.systemid CLASS(VMCMD) ID(user or group) ACCESS(READ)
   ```

   Note: If you do not give users an access authority of READ when creating the profile's access list, the users will not be able to use the command.

3. If the VMCMD class is not already active, use the SETROPTS command to activate it:

   ```
   SETROPTS CLASSACT(VMCMD)
   ```

To allow all class A and B virtual machines to issue the XAUTOLOG command with the ON operand for any target user on any system, enable generic command processing and create a generic VMCMD profile by entering this command:

```
RDEFINE VMCMD XAUTOLOG.ON.** UACC(READ)
```

For information about the XAUTOLOG command when the SECLABEL class is active, see "Security Label Considerations for Profiles in the VMCMD Class" on page 143 and "LOGON, AUTOLOG, and XAUTOLOG Commands" on page 262.

For information about controlling access to terminal devices, see "Protecting Terminals on z/VM" on page 165.

Protecting the FOR Command

The CP FOR command can be used to issue a CP command under the authority of a different user ID, which must be logged on at the time the FOR command is issued. For example, to issue a LINK command under the authority of OTHERID, you can issue:

```
FOR OTHERID CMD LINK USER2 191 777 RR
```

CP defines two versions of the FOR command. The G class version is issued when you are defined as the secondary console of the target user ID in the CP directory. The C class version is issued when you are not defined as the secondary console in the CP directory. The G and C versions are separately controllable and auditable in RACF. However, since the CP directory is not consulted when RACF is protecting the FOR command (unless RACF defers the decision to CP), it makes sense to treat them the same way. Therefore, IBM recommends that if you audit or control one version of the command, you do so for the other version as well.

The ability to perform commands under another user ID is protected by the same resource that allows you to LOGON BY to that user ID. This ability is controlled by profiles in the SURROGAT class. For example, in the previous example, the command issuer would require READ access to LOGONBY.OTHERID in the

SURROGAT class. Note that authorizing a user to issue FOR against a given user ID also authorizes the user to LOGON BY to that user, and authorizing a user to LOGON BY to a given user ID also authorizes the user to issue FOR against that user.

For information on defining SURROGAT profiles, see "Defining Shared User IDs" on page 79.

Use the following procedure to protect the CP FOR command:

1. Define profiles in the SURROGAT class, and permit the appropriate users or groups, as described in "Defining Shared User IDs" on page 79.
2. If the SURROGAT class is not already active, use the SETROPTS command to activate it:

   ```
   SETROPTS CLASSACT(SURROGAT)
   ```

3. Make sure protection for the FOR command is active. (See Chapter 8, "Protecting z/VM with VMXEVENT Profiles," on page 127.) It is on by default if an installation has not changed the setting in the currently active VMXEVENT profile by issuing, for example:

   ```
   RDEFINE VMXEVENT profile-name ADDMEM(FOR.C/NOCTL FOR.G/NOCTL)
   ```

   If protection of the command name is not currently active, activate it by issuing:

   ```
   RALTER VMXEVENT profile-name DELMEM(FOR.C/NOCTL FOR.G/NOCTL)
   SETEVENT REFRESH profile-name
   ```

Note: If the FOR command is not controlled, CP will grant access based on its native security mechanisms. If the issuing user has privilege class C, or the user is on the LOGONBY statement of the target user ID's CP directory entry, access will be granted.

Security Label Considerations for the FOR Command

When the SECLABEL and VMMAC classes are active, z/VM calls RACF to perform a security label check between the command issuer and the target user ID. RACF performs a read/write SECLABEL check between the command issuer's SECLABEL and the SECLABEL of the target user. This requires that the SECLABELs be equivalent. Note that since both users must be logged on, the SECLABELs which are compared are the ones currently assigned to the users' LOGON sessions. These may not be the users' default SECLABELs, since users can LOGON at different security labels, assuming they have the authority to do so.

Protecting the RAC Command Processor and the RACF Command Session

Two profiles in the VMCMD class, RACF and RAC, protect access to the RACF command session, RACFISPF, the RAC command processor, and the RACF panels. Use these two profiles to limit or to allow access on your system:

- Use the RACF profile to protect the use of the RACF command session and RACFISPF.
- Use the RAC profile to protect the use of the RAC command processor and the RACF panels.

If you have both the RACF and the RAC profiles set to a UACC of NONE, and if you have not permitted anyone through an access list, no one will be able to use any of the RACF commands or panels. An absence of one of the profiles means that everyone is permitted to use what that profile was protecting. For each of the profiles, RACF and RAC, READ access permits users to use the commands or panels.

Note: Your installation may require users to supply their logon passwords to enter the RACF command session and RACFISPF. If users choose to change their passwords at this time, and are then denied access to the RACF command session because of a RACF profile in place, their password change is still in effect.

Examples:

1. Create a profile in the VMCMD class:

```
RDEFINE  VMCMD  RACF  UACC(NONE)
```

   or

```
RDEFINE  VMCMD  RAC   UACC(NONE)
```

   Note: If the RACF or RAC profiles do not exist, all users can use a RACF command session and RACFISPF, or RAC and the panels, respectively.

2. Allow selected users to use the respective functions:

```
PERMIT  RACF  CLASS(VMCMD) ID(user or group) ACCESS(READ)
```

   or

```
PERMIT  RAC  CLASS(VMCMD) ID(user or group) ACCESS(READ)
```

   Note: Users of the BLKUPD utility must use the RACF command session. Therefore you should give them access to this profile.

3. Activate the VMCMD class:

```
SETROPTS CLASSACT(VMCMD)
```

Protecting the DIAGNOSE X'88' Subcodes

DIAGNOSE X'88' is an interface whereby an authorized application can validate the password of a user, called the *agent*, and optionally check to see if the agent can work on behalf of another user, called the *target*. By controlling the DIAG088 event in your VMXEVENT profile, you can protect the ability to issue DIAGNOSE X'88' using RACF. DIAGNOSE X'88' is protected by the DIAG088 resource in the VMCMD class. All of the DIAGNOSE X'88' subcodes will call RACF when DIAG088 is being controlled.

For details on the functions provided by DIAGNOSE X'88', see *z/VM: CP Programming Services*.

If DIAG088 is not controlled by RACF or if the VMCMD class is not active or the DIAG088 resource is not defined, the ability to issue DIAGNOSE X'88' is controlled by the OPTION DIAG88 statement in the diagnose issuer's CP directory entry.

For DIAGNOSE X'88' subcode 8, there is an additional RACF check if a target user ID is provided in the DIAGNOSE X'88' parameter list. This check determines whether the agent user can work on behalf of the target user. This ability is controlled by the LOGONBY./*target* profile in the SURROGAT class, for which the agent must have at least READ access. See “Defining Shared User IDs” on page 79 for instructions on using profiles in the SURROGAT class.

Important: Note that by controlling DIAG088, you are protecting the ability to issue DIAGNOSE X'88' with a VMCMD class resource in RACF, rather than by the OPTION DIAG88 statement in the CP directory. However, for subcode 8, CP will call RACF unconditionally for the actual password validation, and for the surrogate check if a target user ID was supplied. That is, even if DIAG088 is not controlled in your VMXEVENT profile, RACF is still called to validate the password and perform the surrogate check for subcode 8.

To protect DIAGNOSE X'88', perform the following steps:

1. Define the diagnose as a profile in the VMCMD class and specify a UACC of NONE:

```
RDEFINE VMCMD DIAG088 UACC(NONE)
```

2. Allow the appropriate users or groups to use the command by giving them READ access:

```
PERMIT DIAG088 CLASS(VMCMD) ID(user or group) ACCESS(READ)
```

3. If the VMCMD class is not already active, use the SETROPTS command to activate it:

```
SETROPTS CLASSACT(VMCMD)
```

4. Make sure control for DIAG088 is active. For more information, see Chapter 8, “Protecting z/VM with VMXEVENT Profiles,” on page 127. It is controlled by default if your installation has not changed the setting in the currently active VMXEVENT profile by issuing, for example:

```
REDEFINE VMXEVENT profile-name ADDMEM(DIAG088/noctl)
```

If protection of the command name is not currently active, activate it by issuing:

```
RALTER VMXEVENT profile-name DELMEM(DIAG088/noctl)
SETEVNET REFRESH profile-name
```

Protecting the DIAGNOSE X'A0' Subcodes

RACF provides six subcodes of DIAGNOSE X'A0' that can be used by application programs. The use of five of the subcodes (X'04', X' 30', X'34', X'3C'and X'50') can be protected using RACF profiles.

RACF profile protection will be in effect and privilege class checking will be ignored for these DIAGNOSE X'A0' subcodes only if all of the following conditions are met:

- Control for DIAG0A0 is turned on.
- A VMCMD class profile has been defined for the subcode being issued.
- The VMCMD class is active.

If any one or more of the above conditions is not met, privilege class checking is enforced.

Table 19. DIAGNOSE A0 Subcodes

Subcode	Description	Privilege Class	VMCMD Class Profile Name
X'04' and X'3C'	Use subcode X'04' to verify a user and validate the user's password. Use subcode X'3C' to verify a user and validate the user's password phrase.	One or more of: A B C D E F	DIAG0A0.VALIDATE
X'30'	To query the current SECLABEL of your own user ID.	ANY	None (ANY privilege class **cannot** be protected)
	To query the current SECLABEL of a different user ID.	One or both of: A B	DIAG0A0.QUERYSEC
X'34'	Use subcode X'34' to update the human-readable-label to SECLABEL correlation table (“HR table”) in CP.	One or both of: A B	DIAG0A0.HRTSTORE
X'38'	Use subcode X'38' to obtain the size of, or a copy of the human-readable-label to SECLABEL correlation table (“HR table”).	ANY	None (ANY privilege class **cannot** be protected)

Table 19. DIAGNOSE A0 Subcodes (continued)			
Subcode	**Description**	**Privilege Class**	**VMCMD Class Profile Name**
X'50'	Use subcode X'50' to retrieve RACF configuration information.	One or both of: A B	DIAG0A0.RACONFIG

You cannot use the CP MODIFY command or statement to change the privilege class of DIAGNOSE X'A0' because DIAGNOSE X'A0' is defined to z/VM as a privilege class ANY diagnose code. Also, if you allow a user ID a specific privilege class, the user ID can perform all tasks at that privilege class level; you may not want to grant the user that level of privilege. For these reasons, it is recommended that you give user IDs access to the VMCMD class profiles rather than give them the required privilege class.

Note: When DIAGNOSE X'A0' control is in effect, regardless of profile existence, RACF is called each time **any** of the protectable subcodes is used; performance impacts are related to the frequency with which the user ID uses the subcode.

Setting Up RACF Protection for a Subcode

To set up protection for one of the three subcodes:

1. Define the subcode as a profile in the VMCMD class using a *profile_name* from Table 19 on page 148 and specifying a UACC of NONE:

   ```
   RDEFINE  VMCMD  profile_name  UACC(NONE)
   ```

2. Allow the appropriate users or groups to use the subcode by giving them READ access to the appropriate profile:

   ```
   PERMIT profile_name CLASS(VMCMD) ID(user
   or group) ACCESS(READ)
   ```

   Note that if you do not give users an access authority of at least READ when creating the access list, the users will not be able to use the subcode.

3. If the VMCMD class is not already active, use the SETROPTS command to activate it:

   ```
   SETROPTS CLASSACT(VMCMD)
   ```

4. Make sure protection of DIAG0A0 is active. It is active by default if an installation has not changed the setting in the currently active VMXEVENT profile by issuing, for example:

   ```
   RDEFINE VMXEVENT EVENTS1 ADDMEM(DIAG0A0/NOCTL)
   ```

   If protection of DIAG0A0 is not currently active, activate it by issuing:

   ```
   RALTER VMXEVENT EVENTS1 DELMEM(DIAG0A0/NOCTL)
   SETEVENT REFRESH EVENTS1
   ```

Remote APPC Connections and Their Effects on RACF for z/VM

Using the APPCPWVL Event to Verify Passwords on APPC CONNECT

When SECURITY(PGM) is specified on an APPC CONNECT to the local system, a user ID and password combination is supplied for verification. If the APPCPWVL event is being controlled in your currently active VMXEVENT profile, z/VM will call RACF to verify that the password is correct for the specified user ID. If APPCPWVL is not being controlled, z/VM will verify the user ID and password combination against information in the CP directory.

When RACF is called to check the password, the user's count of successive incorrect passwords will be incremented if the password is not correct. If the limit established on SETROPTS(PASSWORD(REVOKE)) is reached, the user will be revoked by RACF on the local system.

Considerations When the SECLABEL and VMMAC Classes Are Active

When RACF is active, z/VM may call RACF for an APPC connection (z/VM event APPCCON), which originates from a remote system. If the SECLABEL and VMMAC classes are active, the remote user ID that originated the connection must be defined to RACF. It must also have a user profile in the RACF database.

For example, if user SYSAUSER on System A issued an APPC connection to SYSBUSER on System B and the SECLABEL and VMMAC classes were active, the user ID SYSAUSER must be defined in the RACF database on System B.

Chapter 10. Protecting z/VM Resources

The following sections describe how to create profiles to protect resources on z/VM systems.

See “Defining Profiles for General Resources” on page 98 for general information.

Protecting Real Devices

You can use RACF to control who can connect to z/VM real devices using profiles in the VMDEV resource class. z/VM may call RACF for an authorization check in the VMDEV class when connecting a real device to a virtual machine for exclusive use or connecting a tape device to a virtual machine for shared use, externally this is triggered by three different events:

- DEDICATE statements in the user directory
- ATTACH command
- GIVE command

Profile Considerations in the VMDEV Class

Decide on the type of profiles you want to create. There are several things to consider.

Generic Profiles

You can use generic profiles for z/VM real devices. If you do so, it is strongly recommended that you issue the following command:

```
SETROPTS GENLIST(VMDEV)
```

This command causes one copy of each generic profile for the VMDEV class to be kept in the RACF service machine.

Changes made to a generic device profile may not take effect until the in-storage copy kept in the RACF service machine is refreshed using the SETROPTS GENERIC(VMDEV) REFRESH command.

Because general users cannot issue the SETROPTS command, using generic device profiles can prevent users from promptly and effectively changing device profiles.

Discrete Profiles

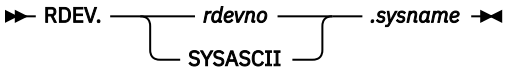
You can use discrete profiles for z/VM real devices. It is recommended that you use discrete profiles for those devices for which you anticipate more frequent profile updates.

Changes made to a discrete device profile immediately affect attempts to connect to the protected device.

Procedure to Protect Real Devices

To protect z/VM real devices, do the following:

1. Profiles in the new VMDEV general resource class have the following format:



where:

RDEV
is a fixed profile prefix

rdevno
is the real device number specified as four hexadecimal digits including leading zeros

SYSASCII
indicates profile used when real ASCII console is to be attached

sysname
is the system to which the profile applies, specified in up to eight alphanumeric character

2. Use the PERMIT command to grant or deny users and groups access to real devices.

```
PERMIT RDEV.0456.* CLASS(VMDEV) ID(SMITH) ACCESS(UPDATE)
```

3. Activate the VMDEV class:

```
SETROPTS CLASSACT(VMDEV)
```

4. Make sure protection of the desired event is active. It is active by default if an installation has not changed the setting in the currently active VMXEVENT profile by issuing, for example:

```
RDEFINE VMXEVENT EVENTS1 ADDMEM(RDEVCTRL/NOCTL)
```

If protection of the desired events is not currently active, activate it by issuing:

```
RALTER VMXEVENT EVENTS1 DELMEM(RDEVCTRL/NOCTL)
SETEVENT REFRESH EVENTS1
```

RACF Real Device Access Authorities

Access authorities correspond to operands that are specified on the CP ATTACH command. The following list shows RACF access authorities and their corresponding CP ATTACH operands:

NONE
Does not allow users to access the device.

Attention:
Anyone who has READ, UPDATE, CONTROL, or ALTER authority to a protected device can copy the data in it. If users copy the data to a device for which they can control the security characteristics, they can potentially downgrade the security characteristics of the copied files. For this reason, you will probably want to assign a UACC of NONE, and then selectively permit a small number of users to access your device, as their needs become known. See *z/VM: RACF Security Server General User's Guide* for information on how to permit selected users or groups to access a device.

READ
Only R/O access allowed

UPDATE
Normal READ/WRITE

CONTROL
Allows the use of SYSCTL operand on the ATTACH command

Note: For a description of the different CP ATTACH operands, refer to *z/VM: CP Commands and Utilities Reference*.

Protecting z/VM Minidisks

You can use RACF to control who can link to z/VM minidisks using profiles in the VMMDISK resource class. z/VM calls RACF for an authorization check in the VMMDISK class for two different events:

1. LINK command—A user's attempt to link to another user's minidisk.

2. MDISK event—A user linking to his or her own minidisk. MDISK events occur at logon time when the user's MDISK directory statements are being processed, or when the user issues a LINK command for a self-owned minidisk.

Public Minidisks

Your installation may have many minidisks that do not contain installation-sensitive data and that are frequently accessed by large numbers of users in READ mode. You can consider using the global access checking table or creating a global minidisk table to help performance. These tables are applicable to public minidisks your installation has identified.

For information on the global minidisk table, see *z/VM: RACF Security Server Macros and Interfaces* and *z/VM: RACF Security Server System Programmer's Guide*.

For information on the global access checking table, see Chapter 7, "Fast Authorization Using the Global Access Checking (GAC) Table," on page 121.

Profile Considerations in the VMMDISK Class

Decide on the type of minidisk profiles you want to create. There are several things to consider.

Generic Profiles

You can use generic profiles for z/VM minidisks. If you do so, it is strongly recommended that you issue the following command:

```
SETROPTS GENLIST(VMMDISK)
```

This command causes one copy of each generic profile for the VMMDISK class to be kept in the RACF service machine.

Changes made to a generic minidisk profile may not take effect until the in-storage copy kept in the RACF service machine is refreshed using the SETROPTS GENERIC(VMMDISK) REFRESH command.

Because general users cannot issue the SETROPTS command, using generic minidisk profiles can prevent users from promptly and effectively changing minidisk profiles. For more information on when changes to minidisk profiles take effect, see "When Changes to Minidisk Profiles Take Effect" on page 308.

Discrete Profiles

You can use discrete profiles for z/VM minidisks. It is recommended that you use discrete profiles for those minidisks for which you anticipate more frequent profile updates.

Changes made to a discrete minidisk profile immediately affect attempts to LINK to the protected minidisk.

ACIGROUP Considerations

If a user's directory entry has an ACIGROUP control statement, the profile names for the user's minidisks are prefaced with the user's ACIGROUP name. See "Use of ACIGROUP Control Statements" on page 94.

Procedure to Protect Minidisks

To protect z/VM minidisks, do the following:

1. Create profiles to protect minidisks:

   ```
   RDEFINE VMMDISK userid.virtual-address
   ```

   where:

   userid
   is the owning user ID.

virtual-address
is the virtual address of the minidisk as defined in the user's CP directory entry

You can define a minidisk with a 4-character virtual address; however, RACF does not allow the first character to be 0. For example, RACF allows SMITH.191, SMITH.1234, and SMITH.002, but does not allow SMITH.0191. To create a minidisk profile for SMITH's A-disk, which would have a virtual address of 191, use the following command:

```
RDEFINE VMMDISK   SMITH.191  UACC(NONE)
```

2. Use the PERMIT command to grant or deny users and groups access to minidisks. Ensure that the owner of the minidisk is included in the access list so that, when the users link to their own minidisks—either with the LINK command or while logging on—the authorization will not fail.

```
PERMIT SMITH.191 CLASS(VMMDISK) ID(SMITH) ACCESS(ALTER)
```

For advice on what authorization to request in the PERMIT command, see "Access Authority for Minidisks on z/VM" on page 154.

Note: Another way to allow every user to have ALTER access to minidisks that begin with each user's own user ID(s) is to use Global Access Checking for the VMMDISK class, and to use the resource name &RACUID.*. For example, after defining the VMMDISK class to the Global Access Checking table, the following command allows every user to have ALTER access to minidisks that begin with each user's own user's ID(s):

```
RALTER GLOBAL VMMDISK ADDMEM(&RACUID.*/ALTER)
```

For more information, see Chapter 7, "Fast Authorization Using the Global Access Checking (GAC) Table," on page 121.

3. Activate the VMMDISK class:

```
SETROPTS CLASSACT(VMMDISK)
```

4. Make sure protection of LINK and MDISK is active. It is active by default if an installation has not changed the setting in the currently active VMXEVENT profile by issuing, for example:

```
RDEFINE VMXEVENT EVENTS1 ADDMEM(LINK/NOCTL MDISK/NOCTL)
```

If protection of LINK and MDISK is not currently active, activate it by issuing:

```
RALTER VMXEVENT EVENTS1 DELMEM(LINK/NOCTL MDISK/NOCTL)
SETEVENT REFRESH EVENTS1
```

Access Authority for Minidisks on z/VM

Minidisks on z/VM can have one of the following access authorities:

NONE
Does not allow users to access the minidisk.

Attention
Anyone who has READ, UPDATE, CONTROL, or ALTER authority to a protected minidisk can copy the data in it. If users copy the data to a minidisk for which they can control the security characteristics, they can potentially downgrade the security characteristics of the copied files. For this reason, you will probably want to assign a UACC of NONE, and then selectively permit a small number of users to access your minidisk, as their needs become known. See *z/VM: RACF Security Server General User's Guide* for information on how to permit selected users or groups to access a minidisk.

READ
Allows users to read from the minidisk. This enables users to request any read-only link mode on the CP LINK command. Read-only link modes include R, RR, SR, and ER. (Note that users who can read files on a minidisk can copy or print them.)

UPDATE
Allows users to read from, or write to, the minidisk. This enables users to request any of the read-only and some of the write link modes on the CP LINK command. The allowed write link modes include W, WR, SW, and EW.

CONTROL
Allows users to read from, or write to, the minidisk. This enables users to request any of the read-only link modes and all of the write link modes except MW on the CP LINK command. In addition to the link modes allowed for READ and UPDATE access, users may request a link mode of M, MR, or SM.

ALTER
Allows users to read from, or write to, the minidisk. This enables users to request any valid link mode on the CP LINK command, including MW (multiwrite).

Unlike other general resource classes, ALTER access to a discrete VMMDISK profile does not, by itself, allow a user to read, alter, or delete the profile, or to modify its access list.

As an alternative approach to allow users to manage VMMDISK profiles, you can create a group to own the profiles and connect users to that group with the SPECIAL attribute. For example, to enable users USERA and USERB to manage VMMDISK profiles for USER1.191, use the following RACF commands:

```
ADDGROUP ADMVMMD
RALTER VMMDISK USER1.191 OWNER(ADMVMMD)
CONNECT (USERA USERB) GROUP(ADMVMMD) SPECIAL
```

Users with ALTER access to a generic VMMDISK profile have no authority over the profile itself.

Note: For a description of the different CP LINK access modes, refer to *z/VM: CP Commands and Utilities Reference*.

SYSSEC Considerations for LINK and MDISK

The SYSSEC macro, coded in the RACF module HCPRWA, can affect the final outcome of resource requests in the VMMDISK class. The SYSSEC macro defines the relationship between RACF's response to a resource access request and the final disposition of that request by z/VM. This relationship changes as you change the SYSSEC statement in HCPRWA.

For example, a user may link to a minidisk which is not protected by a RACF profile. SYSSEC can be coded so that for this case, RACF will do one of the following:

- Allow access
- Disallow access
- Defer the access decision to z/VM

You should know how the SYSSEC macro is coded on your system. See *z/VM: RACF Security Server Macros and Interfaces* for more information.

Security Label Checking for LINK and MDISK Events

When the SECLABEL class is active, z/VM calls RACF for authorization checking for a link to a minidisk even if control is turned off for the LINK and MDISK events.

If the SECLABEL class is active, but you do not want to use RACF minidisk protection, deactivate the VMMDISK class by issuing:

```
SETROPTS NOCLASSACT(VMMDISK)
```

z/VM still calls RACF for MAC authorization, but RACF defers the authorization to z/VM. z/VM will then perform LINK authorization based on the CP directory.

The type of SECLABEL authorization required is based on the link mode the user requested for the minidisk.

1. If the user is requesting a link mode that requires READ access to the VMMDISK profile, a read-only authorization request is performed.
2. If the user is requesting a link mode that requires UPDATE, CONTROL, or ALTER access to the VMMDISK profile, a read/write authorization request is performed.

Note: When the SECLABEL class is active and SETROPTS MLS(FAILURES) is in effect, RACF *downgrades* MDISK requests for MR or WR access if the SECLABEL of the user requesting the link does not equal the SECLABEL of the minidisk. In this case, the user receives read-only access to the minidisk, but only if so authorized.

The outcome of the request depends on the SETROPTS options that are in effect. For more information, see "Security Label Authorization Checking" on page 313.

Additionally, when the SECLABEL class is active and SETROPTS MLACTIVE(FAILURES) has been issued, profiles in the VMMDISK class are required to contain a SECLABEL. Therefore, authorization requests for a minidisk which does not have an assigned SECLABEL will fail.

Protecting Virtual Unit Record Devices

You can use RACF to control which users can send files to virtual unit record devices using profiles in the VMRDR resource class. Virtual unit record devices are the readers, punches, and printers for any virtual machine.

When protecting virtual unit record devices, RACF is called for authorization for several events. When control is turned on for the TRANSFER.G command, the other commands that are protected include:

- TRANSFER and CHANGE TO (for spool files you own or originated)
- CLOSE TO
- SPOOL TO
- SPOOL FOR
- VMDUMP TO
- DIAG094 (DIAGNOSE code X'94' with the TO parameter)

The commands protected by RACF when control is turned on for TRANSFER.D are:

- TRANSFER and CHANGE TO (for spool files you do not own and did not originate)
- TRSAVE TO

You can use this process to control who can send files for processing by batch machines or networking machines.

Note:

1. RACF protects all unit record devices for a particular virtual machine in the same manner. For example, if USERA can send files to USERB's reader, USERA can also send files to USERB's punch and printer. Conversely, if USERA cannot send files to USERB's reader, USERA also cannot send files to USERB's punch or printer.
2. While installing RACF, your installation can create a VMRDR profile for each user whose virtual reader is defined in the z/VM directory. As you add new users to the system, you might need to manually create VMRDR profiles for them. (For more information on defining user IDs during installation, see *z/VM: RACF Program Directory*.)

To define and protect a user's virtual unit record devices with RACF, take the following steps:

1. Create a profile in the VMRDR class:

   ```
   RDEFINE  VMRDR  userid  UACC(NONE)
   ```

   where *userid* is the owning user ID.

Note: If a user's directory entry has an ACIGROUP control statement, the profile names for the user's virtual unit record devices must be prefaced with the user's ACIGROUP name. See "Use of ACIGROUP Control Statements" on page 94.

2. To allow users to send files to readers, printers, and punches protected by the profile, give them UPDATE access to the profile:

```
PERMIT  userid  CLASS(VMRDR)  ID(user or group)  ACCESS(UPDATE)
```

3. Activate the VMRDR class:

```
SETROPTS CLASSACT(VMRDR)
```

4. Make sure protection of TRANSFER.D and TRANSFER.G is active. It is active by default if an installation has not changed the setting in the currently active VMXEVENT profile by issuing, for example:

```
RDEFINE VMXEVENT EVENTS1 ADDMEM(TRANSFER.D/NOCTL TRANSFER.G/NOCTL)
```

If protection of TRANSFER.D and TRANSFER.G is not currently active, activate it by issuing:

```
RALTER VMXEVENT EVENTS1 DELMEM(TRANSFER.D/NOCTL TRANSFER.G/NOCTL)
SETEVENT REFRESH EVENTS1
```

Attention:
You should be aware that, because these commands are frequently used, controlling their use can have a significant effect on system performance.

SYSSEC Considerations for Unit Record Devices

The SYSSEC macro, coded in the RACF module HCPRWA, can affect the final outcome of resource requests in the VMRDR class. The SYSSEC macro defines the relationship between RACF's response to a resource access request and the final disposition of that request by z/VM. This relationship changes as you change the SYSSEC statement in HCPRWA.

For example, a user may send a file to a virtual unit record device which is not protected by a RACF profile. SYSSEC can be coded so that for this case, RACF will do one of the following:

- Allow access
- Disallow access
- Defer the access decision to z/VM

You should know how SYSSEC is affecting your VMRDR requests. See *z/VM: RACF Security Server Macros and Interfaces* for more information.

Security Label Considerations for Unit Record Devices

If a security label exists in the VMRDR profile protecting the resource, and the SECLABEL class is active, RACF compares the security label of the user with the security label contained in the profile. RACF grants access if the security label of the user is equal to or higher than the security label contained in the profile.

Protecting RSCS Nodes

You can use RACF to control which users can send files to specific RSCS nodes using the TAG DEVICE or TAG FILE commands.

To control access to nodes, do the following:

1. Use the RDEFINE command to define a profile for the node:

```
RDEFINE  VMNODE  node-id  UACC(NONE)
```

where *node-id* is the node id defined in the RSCS service machine.

2. Use the PERMIT command to give appropriate users and groups UPDATE access authority to the profile:

```
PERMIT  node-id  CLASS(VMNODE)  ID(user or group)  ACCESS(UPDATE)
```

   UPDATE access authority is required for a user to send files to the node.

3. Activate the VMNODE class:

```
SETROPTS CLASSACT(VMNODE)
```

4. Make sure protection of TAG is active. It is active by default if an installation has not changed the setting in the currently active VMXEVENT profile by issuing, for example:

```
RDEFINE VMXEVENT EVENTS1 ADDMEM(TAG/NOCTL)
```

   If protection of TAG is not currently active, activate it by issuing:

```
RALTER VMXEVENT EVENTS1 DELMEM(TAG/NOCTL)
SETEVENT REFRESH EVENTS1
```

Attention:
You should be aware that, because the TAG command is frequently used, controlling its use can have a significant effect on system performance.

SYSSEC Considerations for RSCS Nodes

The SYSSEC macro, coded in the RACF module HCPRWA, can affect the final outcome of resource requests in the VMNODE class. The SYSSEC macro defines the relationship between RACF's response to a resource access request and the final disposition of that request by z/VM. This relationship changes as you change the SYSSEC statement in HCPRWA.

For example, a user may send a file to an RSCS node which is not protected by a RACF profile. SYSSEC can be coded so that for this case, RACF will do one of the following:

- Allow access
- Disallow access
- Defer the access decision to z/VM

You should know how SYSSEC is affecting your VMNODE requests. See *z/VM: RACF Security Server Macros and Interfaces* for more information.

Security Label Considerations for RSCS Nodes

If a security label exists in the VMNODE profile protecting the resource, and the SECLABEL class is active, RACF compares the security label of the user with the security label contained in the profile, and grants access if the security label of the user is equal to or higher than the security label contained in the profile.

Additionally, when the SECLABEL class is active, an authorization check is made in the VMMAC class for the TAG FILE command. See “Protecting z/VM Events with the VMMAC Class” on page 264 for more information.

To exclude RSCS from TAG and TRANSFER checking, create an individual VMXEVENT profile that turns off control of the SPOOL and TAG commands for RSCS:

```
RDEFINE VMXEVENT USERSEL.RSCS
RALTER VMXEVENT USERSEL.RSCS ADDMEM(TAG/NOCTL TRANSFER.G/NOCTL TRANSFER.D/NOCTL)
SETEVENT REFRESH USERSEL.RSCS
```

Note: You will also need to assign RSCS a seclabel of SYSNONE to handle traffic for any network or user classification.

Protecting Alternate User IDs

The alternate user ID function is a z/VM facility that enables one virtual machine to act with the access authority of another virtual machine. This relationship is established by use of the z/VM Diagnose code X'D4'. In most cases, this diagnose code is issued by a batch virtual machine (hereafter referred to as the “master”). This allows a batch processor (or “worker”) to perform work for an end user (or “alternate user”).

You can use RACF to protect which workers will be able to perform work on behalf of a particular alternate user through the use of profiles in the VMBATCH resource class.

The ICHRCX02 RACHECK postprocessing exit allows an alternate user to access data that the worker can access, but the alternate user cannot normally access. It can be used, for example, to allow users to access a restricted compiler only when submitting a batch job to a specified batch machine. This exit is shipped with the RACF product but is not enabled. If this RACHECK postprocessing exit is desirable, enable the ICHRCX02 exit (see *z/VM: RACF Security Server System Programmer's Guide* for more information on this procedure).

To allow the use of alternate user IDs, take the following steps:

1. To allow a new user to use the alternate user ID function, create a profile for that user in the VMBATCH class as follows:

   ```
   RDEFINE  VMBATCH  userid  UACC(NONE)
   ```

   Note: While installing RACF, your installation can create a VMBATCH profile for each user defined in the z/VM directory. As you add new users to the system, you define their user IDs to RACF in the VMBATCH resource class if those users intend to use the alternate user ID function. (For more information on defining user IDs during installation, see *z/VM: RACF Program Directory*.)

2. Authorize that user ID to be an alternate user ID for a worker machine in one of the following ways:

   - Use the PERMIT command to give the worker CONTROL access authority to the VMBATCH profile for the alternate user ID:

     ```
     PERMIT  userid  CLASS(VMBATCH)  ID(worker)  ACCESS(CONTROL)
     ```

     For example, to allow worker batch machine BAT1 to perform work on behalf of user USER1, issue the following command:

     ```
     PERMIT  USER1  CLASS(VMBATCH)  ID(BAT1)  ACCESS(CONTROL)
     ```

     This allows USER1 to submit batch jobs either to worker batch machine BAT1 directly, or to the master batch machine, who in turn would assign the job to worker machine BAT1.

   - You can also organize worker batch machines into groups and authorize them all to be able to perform work on behalf of an alternate user. Take the following steps:

     a. Define a group for the worker machines. For example, the following command defines group BATGROUP:

        ```
        ADDGROUP  BATGROUP
        ```

     b. Connect each worker machine to the group. For example, the following command connects the worker machines BAT1, BAT2, and BAT3 to the group BATGROUP:

        ```
        CONNECT  (BAT1  BAT2  BAT3)  GROUP(BATGROUP)
        ```

     c. Allow these workers to perform work on behalf of the alternate user:

        ```
        PERMIT alt-userid CLASS(VMBATCH) ID(BATGROUP) ACCESS(CONTROL)
        ```

d. To ensure access will be granted with this method, you must have either list-of-groups access checking activated:

```
SETROPTS GRPLIST
```

or change the alternate user's default group to match the group name you specified in the previous PERMIT example:

```
ALTUSER alt-userid DFLTGRP(BATGROUP)
```

3. Activate the VMBATCH class:

```
SETROPTS CLASSACT(VMBATCH)
```

4. Make sure protection of DIAG0D4 is active. It is active by default if an installation has not changed the setting in the currently active VMXEVENT profile by issuing, for example:

```
RDEFINE VMXEVENT EVENTS1 ADDMEM(DIAG0D4/NOCTL)
```

If protection of DIAG0D4 is not currently active, activate it by issuing:

```
RALTER VMXEVENT EVENTS1 DELMEM(DIAG0D4/NOCTL)
SETEVENT REFRESH EVENTS1
```

Security Label Considerations for Alternate User IDs

When the SECLABEL class is active (even if control has been turned off for the DIAG0D4 event using a VMXEVENT profile and the SETEVENT command), RACF verifies that both the worker machine and the alternate user ID have access to the specified SECLABEL, which appears in the parameter list for DIAGNOSE X'D4', subcode X'04'. However, these RACF checks do not ensure that the worker machine is running at a SECLABEL equivalent to the alternate user ID's SECLABEL.

To enforce SECLABEL checking, an installation should have a worker batch machine available for each SECLABEL used by alternate user IDs, and the worker batch machine should only have authorization to that one SECLABEL. The SECLABEL of the work done by each worker machine should be equivalent to the SECLABEL at which the alternate user is logged on.

When the SECLABEL class is active, z/VM does not allow DIAG0D4 subcode X' 00'.

RACF compares the SECLABEL of the worker machine with the SECLABEL contained in the VMBATCH profile when all of the following conditions are true:

- The SECLABEL class is active
- Control is on for the DIAG0D4 event (it is on by default unless a VMXEVENT profile is active containing a DIAG0D4/NOCTL member)
- A SECLABEL exists in the VMBATCH profile for the alternate user ID.

RACF grants access if the SECLABEL of the worker machine is equal to or higher than the SECLABEL contained in the VMBATCH profile.

Allowing Batch Machines to Access a User's Minidisks

When a worker machine is running a user's job, and the job attempts to link to the user's minidisk, z/VM calls RACF for a LINK authorization check in the VMMDISK class. Note that this is a LINK request and not a self-link (or MDISK) since it is the worker machine issuing the LINK. Since the worker machine has the user's access authorities, the user must have the correct access authority to the minidisk (usually ALTER), or the LINK will fail.

For example, user SMITH can do the following:

```
PERMIT SMITH.191 CLASS(VMMDISK) ID(SMITH) ACCESS(ALTER)
```

As security administrator, you can create an entry in the global access checking table that does the equivalent of the preceding PERMIT for all users on the system:

```
RALTER GLOBAL VMMDISK ADDMEM(&racuid.*/ALTER)
```

For more information on global access checking, see Chapter 7, “Fast Authorization Using the Global Access Checking (GAC) Table,” on page 121.

Protecting Restricted Shared Memory Segments

RACF can protect named saved segments (NSS) or discontiguous saved segments (DCSS) that are defined with the RSTD option of the CP DEFSEG or CP DEFSYS commands. To determine if your installation has any NSS's or DCSS's defined as restricted, issue QUERY NSS from a privilege class E user ID. Those segments for which an R is displayed under the CL column are defined as restricted. If you need to define restricted segments, refer to the DEFSEG or DEFSYS commands in *z/VM CP Command and Utility Reference*.

The segments are protected by defining RACF profiles in the VMSEGMT class. z/VM calls RACF for authorization checking on a restricted segment when one of the following events occurs:

- A user IPLs a restricted segment.
- DIAGNOSE code X'64' is issued to find or load a restricted segment. This includes subcodes X'0000', X'0004', X'000C', X' 0010', and X'0018'.
- A user attempts to start a trace file (TRSOURCE ENABLE) for a VMGROUP. (This call is made for **all** named saved segments and **only** when the SECLABEL class is active.)

To use this protection for restricted segments with the VMSEGMT class, follow these steps:

1. Define RACF profiles for the segments to be protected.
   - For an NSS, the profile naming convention is *NSS.spacename*
   - For a DCSS, the profile naming convention is *DCSS.spacename*

   where *spacename* is the name of the space in which the segment resides and is specified when the segment is defined on the CP DEFSEG or DEFSYS command. A space can contain many DCSSs, but the entire space is loaded. In the case of an NSS, there is only one NSS per space.

   ```
   RDEFINE  VMSEGMT  profile_name UACC(NONE)
   ```

2. Authorize the appropriate users to use the restricted segment.

   ```
   PERMIT profile_name CLASS(VMSEGMT)
          ID(user or group) ACCESS(access_authority)
   ```

   UPDATE authority is required if a user must have shared write access to a DCSS or NSS; otherwise, READ authority is sufficient. In other words, if a DCSS or NSS is defined with at least one segment that has the SW or SN attribute, then UPDATE access is required. To start a trace file (TRSOURCE ENABLE) for a VMGROUP, a user must have UPDATE access to the VMSEGMT profile which protects the NSS associated with the VMGROUP.
3. Activate the VMSEGMT class.

   ```
   SETROPTS CLASSACT(VMSEGMT)
   ```

4. Make sure protection of RSTDSEG is active. It is active by default if an installation has not changed the setting in the currently active VMXEVENT profile by issuing, for example:

   ```
   RDEFINE VMXEVENT EVENTS1 ADDMEM(RSTDSEG/NOCTL)
   ```

If protection of RSTDSEG is not currently active, activate it by issuing:

```
RALTER VMXEVENT EVENTS1 DELMEM(RSTDSEG/NOCTL)
SETEVENT REFRESH EVENTS1
```

Security Label Considerations in the VMSEGMT Class

When the SECLABEL class is active, z/VM calls RACF for authorization checking for restricted segments even if control is turned off for the RSTDSEG event.

If your installation does not want to use SECLABEL checking for the VMSEGMT class, deactivate the class by issuing:

```
SETROPTS NOCLASSACT(VMSEGMT)
```

z/VM still calls RACF for MAC authorization, but RACF defers the authorization to z/VM. z/VM treats this as a successful MAC authorization for the RSTDSEG event or TRSOURCE ENABLE event.

If the SECLABEL class is active and the SETROPTS MLACTIVE(FAIL) option is in effect, you must assign security labels to each of your VMSEGMT profiles by issuing:

```
RALTER VMSEGMT profile-name SECLABEL(seclabel-name)
```

Because segments are protected based on the space name, you should ensure all segment data within a space is at the same SECLABEL.

The type of SECLABEL authorization required is based on the RACF access mode required for accessing the restricted segment:

- For segments that require READ access to the VMSEGMT profile, a read-only authorization request is performed.
- For segments that require UPDATE access to the VMSEGMT profile, a read/write authorization request is performed.

The outcome of the request depends on the SETROPTS options that are in effect. For more information, see “Security Label Authorization Checking” on page 313.

Additionally, when the SECLABEL class is active and SETROPTS MLACTIVE(FAILURES) has been issued, profiles in the VMSEGMT class are required to contain a SECLABEL. Therefore, authorization requests for a segment which does not have an assigned SECLABEL will fail.

Protecting Guest LANs and Virtual Switches

RACF can be used to protect Guest LANs and virtual switches using profiles in the VMLAN class. From an access control perspective, guest LANs and virtual switches are treated the same way. In this document, the term "LAN" will be used to mean both guest LANs and virtual switches. When differences exist, they will be identified. For more information on Guest LANs and virtual switches, see *z/VM Connectivity*.

The VMLAN class contains two sets of profiles to protect LANs; base profiles that control the ability of a z/VM user to use a LAN, and, for IEEE VLAN-aware virtual switches, VLAN ID-qualified profiles that are used to assign a user to one or more IEEE VLANs.

Base Profiles

Base profiles are named *userid.name* where *userid* is the LAN owner and *name* is the name of the LAN. Both qualifiers will be eight characters or less. For example, guest LAN NET130 owned by TOMBRADY is represented by profile TOMBRADY.NET130. In the case of a virtual switch, *userid* will always be "SYSTEM". These profiles will control authorization and auditing of attempts by any user to COUPLE to a guest LAN or virtual switch. A user must have UPDATE access to the profile in order for the COUPLE to be authorized.

Promiscuous Mode Authorization

Promiscuous Mode is a mode of operation where the network adapter intercepts all data flowing over the network regardless of destination MAC or IP address. It is sometimes referred to as "LAN sniffing". This mode allows troubleshooting of potential networking problems using existing network debug tools such as **tcpdump** or **ethereal**. Since Promiscuous Mode in a guest virtual machine environment provides the guest NIC with a copy of all data traversing the LAN segment, authorization to use Promiscuous Mode should be given only when necessary. A user must be given CONTROL access to the base profile in order for Promiscuous Mode to be authorized. See *z/VM Connectivity* for more information about using Promiscuous Mode to trouble shoot virtual networking problems.

VLAN ID-Qualified Profiles

There are now two types of virtual switches - user based and port based. User based are the default, and will continue to work with RACF as in previous releases. Access to the virtual switch is on a user id basis. All ports for a guest have the same attributes and VLAN ids.

For a port based virtual switch, all access is on a port basis. A single guest (user) can have multiple unique ports connected to the same virtual switch instance. Each port has its own attributes (promiscuous and OSDSIM authority) and VLAN ids. The ports must be defined with the CP SET VSWITCH PORTNUMBER command. The attributes and VLAN ids are configured with the CP SET VSWITCH PORTNUMBER and the CP SET VSWITCH VLANID commands.

If a virtual switch is VLAN aware (defined with the "VLAN *defvid*" parameter), then a secondary set of VLAN ID-qualified VMLAN profiles are used to control the ability of a virtual machine to become a member of a particular IEEE VLAN. These profiles are named SYSTEM.*name.vid*, where *name* is the name of the virtual switch and *vid* is a VLAN ID having a value between 1 and 4094, inclusive. The *vid* qualifier must consist of four decimal digits; leading zeroes must be entered for VLAN IDs less than four digits. See below for an example.

A user must have UPDATE access to a qualified profile to be considered authorized. No auditing will be performed as a result of checking the VLAN ID-qualified profiles and no violations will be reported. These profiles are consulted by RACF in order to build a list of authorized VLAN IDs on a given virtual switch for a given user. The list is returned to z/VM. The VLAN ID-qualified profiles are only consulted if the user has UPDATE access to the base profile protecting the virtual switch.

z/VM will use the list of authorized VLAN IDs returned from RACF differently for user based and port based virtual switches. For user based virtual switches, the VLAN ID list returned to z/VM is used to configure all connections for the user. VLANs may be specified in the User Directory entry on a per-user basis. The CP VLAN list is ignored.

For port based virtual switches, a single user may have multiple ports. Each port must be defined and configured via CP or on the NICDEF statement in the user directory. That includes assigning the VLAN IDs to the port. The VLAN ID list returned by RACF will be used for authorization only. If a port is configured (via CP) for VLAN 3, then the RACF list returned must contain VLAN 3, or the connection will fail.

Note that z/VM allows the specification of a default VLAN ID when defining a virtual switch to z/VM. RACF does not have a mechanism for designating a default VLAN ID using profiles in the VMLAN class. If a base profile grants a user access to a VLAN aware virtual switch, but the user is not permitted to any VLAN ID-qualified profiles for that virtual switch, the user will be authorized with no VLAN access. The user will have no connectivity through the virtual switch until access to any VLAN ID-qualified profile is granted.

It is up to the administrator to ensure consistency between a base profile and its set of VLAN ID-qualified profiles. For example, assume a user has been granted UPDATE access to the base profile and to a set of VLAN ID-qualified profiles. If the user's access is removed from the base profile, RACF will not automatically remove the user's access from the VLAN ID-qualified profiles. The VLAN ID-qualified profiles will be ignored for this user.

Note:

1. VLAN ID-qualified profiles must be discrete; generic profiles will be ignored.
2. Global Access Checking cannot be used for VLAN ID-qualified profiles.

Examples

Example 1

Define the VMLAN profile protecting the guest LAN named NET130, owned by TOMBRADY. Allow anyone in the PATRIOTS group to COUPLE to the LAN.

```
RAC RDEFINE VMLAN TOMBRADY.NET130 UACC(NONE)

RAC PERMIT TOMBRADY.NET130 CLASS(VMLAN) ID(PATRIOTS) ACCESS(UPDATE)
```

Note: After issuing the RDEFINEs and PERMITs as shown in Example 1 and Example 2, you must activate the VMLAN class:

```
SETROPTS CLASSACT(VMLAN)
```

Then make sure that protection of COUPLE.G is active. It is active by default if an installation has not changed the setting in the currently active VMXEVENT profile by issuing, for example:

```
RALTER VMXEVENT EVENTS1 ADDMEM(COUPLE.G/NOCTL)
```

If protection of COUPLE.G is not currently active, activate it by issuing:

```
RALTER VMXEVENT EVENTS1 DELMEM(COUPLE.G/NOCTL)
SETEVENT REFRESH EVENTS1
```

Example 2

In this example, an administrator wants to define the RACF profiles to represent the user based virtual switch named SWITCH01. Additionally, the administrator wants to assign user SUSAN to VLAN IDs 5, 10, and 15, WILL to VLAN ID 10, and MARYELEN to VLAN ID 1.

```
RAC RDEFINE VMLAN SYSTEM.SWITCH01 UACC(NONE)

RAC PERMIT SYSTEM.SWITCH01 CLASS(VMLAN) ID(SUSAN WILL MARYELEN)
           ACCESS(UPDATE)

RAC RDEFINE VMLAN SYSTEM.SWITCH01.0001 UACC(NONE)

RAC PERMIT SYSTEM.SWITCH01.0001 CLASS(VMLAN) ID(MARYELEN) ACCESS(UPDATE)

RAC RDEFINE VMLAN SYSTEM.SWITCH01.0005 UACC(NONE)

RAC PERMIT SYSTEM.SWITCH01.0005 CLASS(VMLAN) ID(SUSAN) ACCESS(UPDATE)

RAC RDEFINE VMLAN SYSTEM.SWITCH01.0010 UACC(NONE)

RAC PERMIT SYSTEM.SWITCH01.0010 CLASS(VMLAN) ID(SUSAN WILL)
           ACCESS(UPDATE)

RAC RDEFINE VMLAN SYSTEM.SWITCH01.0015 UACC(NONE)

RAC PERMIT SYSTEM.SWITCH01.0015 CLASS(VMLAN) ID(SUSAN) ACCESS(UPDATE)
```

Example 3

In this example, an administrator wants to define the RACF profiles to represent the port based virtual switch named SWITCH02. The virtual switch has two ports defined for SUSAN - Port 1 is an access port configured to use VLAN ID 5, and port 2 is a trunk port configured to use VLAN IDs 10 and 15. The user WILL has port 3 defined. Port 3 is an access port configured to user VLAN ID 10. The administrator wants to assign VLAN IDs to user SUSAN for ports 1 and 2, and for user WILL for port 3. So SUSAN will be assigned VLAN IDs 5, 10 and 15, and WILL be assigned VLAN ID 10.

```
RAC RDEFINE VMLAN SYSTEM.SWITCH02 UACC(NONE)

RAC PERMIT SYSTEM.SWITCH02 CLASS(VMLAN) ID(SUSAN WILL)
           ACCESS(UPDATE)

RAC RDEFINE VMLAN SYSTEM.SWITCH02.0005 UACC(NONE)
```

```
RAC PERMIT SYSTEM.SWITCH02.0005 CLASS(VMLAN) ID(SUSAN) ACCESS(UPDATE)

RAC RDEFINE VMLAN SYSTEM.SWITCH02.0010 UACC(NONE)

RAC PERMIT SYSTEM.SWITCH02.0010 CLASS(VMLAN) ID(SUSAN WILL)
           ACCESS(UPDATE)

RAC RDEFINE VMLAN SYSTEM.SWITCH02.0015 UACC(NONE)

RAC PERMIT SYSTEM.SWITCH02.0015 CLASS(VMLAN) ID(SUSAN) ACCESS(UPDATE)
```

See the note in Example 1 about activating the VMLAN class.

SYSSEC Considerations for Guest LANs

The SYSSEC macro, coded in the RACF module HCPRWA, can affect the final outcome of resource requests in the VMLAN class. The SYSSEC macro defines the relationship between RACF's response to a resource access request and the final disposition of that request by z/VM. This relationship changes as you change the SYSSEC statement in HCPRWA.

For example, a user may link to a Guest LAN which is not protected by a RACF profile. SYSSEC can be coded so that for this case, RACF will do one of the following:

- Allow access
- Disallow access
- Defer the access decision to z/VM

You should know how the SYSSEC macro is coded on your system. See *z/VM: RACF Security Server Macros and Interfaces* for more information.

Security Label Considerations for Guest LANs

When the SECLABEL class is active, z/VM calls RACF for authorization checking for Guest LANs, even if control is turned off for the COUPLE.G event. The SECLABEL check is performed in concert with the access check, and they cannot be controlled separately when the SECLABEL class is active. RACF performs a read/write SECLABEL check between the user's SECLABEL and the SECLABEL in the VMLAN profile. The outcome of the request depends on the SETROPTS options that are in effect. For more information, see "Security Label Authorization Checking" on page 313.

If your installation does not want to use Guest LAN protection in the VMLAN class when the SECLABEL class is active, deactivate the class by issuing:

```
SETROPTS NOCLASSACT(VMLAN)
```

z/VM still calls RACF for MAC authorization, but RACF defers the authorization to z/VM. z/VM will then perform the COUPLE authorization based on definitions in z/VM.

If the SECLABEL class is active and the SETROPTS MLACTIVE(FAILURES) option is in effect, profiles in the VMLAN class are required to contain a SECLABEL. Therefore, authorization requests for a Guest LAN which does not have an assigned SECLABEL will fail. If a VLAN ID-qualified profile does not have a SECLABEL, then the user will not be authorized to that VLAN ID. All the VLAN ID-qualified profiles should have the same SECLABEL as that of the base profile protecting the virtual switch.

Protecting Terminals on z/VM

There are several methods of controlling the use of terminals connected to your z/VM system. If you create profiles in the TERMINAL or GTERMINL classes, access to certain terminals is protected. You can also choose options that will prevent access to undefined terminals, limit specific users to specific terminals, and restrict access to the system during certain hours and days of the week.

For a description of authorization checking for terminals, see "Using Security Labels to Control Terminals" on page 168.

Creating Profiles in the TERMINAL and GTERMINL Classes

If you create a profile in the TERMINAL or GTERMINL class, users must have at least READ access authority to the profile in order to use the terminal(s) protected by the profile.

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 profile name consists of a prefix concatenated with the real address of the terminal.

   For example, if the prefix is LOGN, for a terminal whose real address is 110B, you would define it as follows:

   ```
   RDEFINE  TERMINAL  LOGN110B  UACC(NONE)
   ```

   Note:

   a. If your system has PVM installed, users might be able to enter the system from a protected terminal by issuing the DIAL PVM command, then logging on. The logon occurs at a "logical terminal", whose address cannot be predicted. You can prevent this by issuing the following command:

      ```
      SETEVENT NODIAL
      ```

      This prevents users from issuing the DIAL PVM command. (Users will still be able to use PVM by logging on, then issuing the PASSTHRU command.)

   b. To determine which address to use when defining a terminal, do one of the following:

      - Go to the terminal itself, and issue the following command before logging on to the system:

        ```
        MESSAGE * Test message
        ```

        The message will indicate the address of the terminal unless you are issuing the message from a VTAM® terminal, in which case you must use the VTAM LU name to define the terminal. (See the next bulleted item.)

        For a discussion of prelogon messages for SETEVENT LIST, refer to *z/VM: RACF Security Server Command Language Reference*.

      - If someone is currently logged on to the terminal, and is not using a *dialed* terminal, issue the following command:

        ```
        QUERY userid
        ```

        The system will indicate the user's terminal address which, in the case of a VTAM terminal, will also be the VTAM LU name.

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 will not be authorized to use the terminal. For example, the following command grants users SMITH and JONES READ access authority to terminal LOGN110B.

   ```
   PERMIT LOGN110B 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 profile.

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.

SETROPTS RACLIST Processing on Shared Systems: RACF does not automatically propagate the SETROPTS command to non-cluster systems sharing the database for RACLIST option or the combination of the RACLIST and REFRESH options. For this combination of options, the SETROPTS command must be issued to each system sharing the database. See “SETROPTS Command Propagation” on page 112

However, if you do not issue the SETROPTS command with the RACLIST option on a system sharing a RACF database and that system needs to re-IPL, RACLIST is performed for that system when a re-IPL occurs.

Creating a Profile in the GTERMINL Class: If you want to protect several terminals in the same way, but their real addresses 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 at addresses 14CC, 35AB, and 20EE with one profile, you could create a profile with a name you choose, such as DEPT35:

```
RDEFINE  GTERMINL  DEPT35  UACC(NONE)
         ADDMEM(LOGN14CC LOGN35AB LOGN20EE)
```

To allow group FINANCE to use these terminals, enter the following:

```
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, whose real address is 26DD, with the same profile, change the DEPT35 profile as follows:

```
RALTER  GTERMINL  DEPT35  ADDMEM(LOGN26DD)
SETROPTS  RACLIST(TERMINAL)  REFRESH
```

To stop protecting the terminal whose real address is 35AB with this profile, change the DEPT35 profile as follows:

```
RALTER  GTERMINL  DEPT35  DELMEM(LOGN35AB)
SETROPTS  RACLIST(TERMINAL)  REFRESH
```

Controlling the Use of Undefined Terminals

You can also use RACF to control the use of undefined terminals 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 the following:

```
SETROPTS  TERMINAL(READ)
```

To prevent undefined terminals from being used for logging on, enter the following:

```
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 will be able to log on to your system.*

Combining the SETROPTS TERMINAL Command with TERMINAL Profiles

If you want to control selected terminals, specify READ. 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 containing 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. (See **Attention** box above.)

Restricting Specific Groups of Users to Specific Terminals

When defining or changing a group profile, you can specify that the group can only log on 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 only log on to terminals in the payroll office, protect the payroll terminals with a profile:

```
RDEFINE  GTERMINL  PAYTERMS  ADDMEM(LOGN1344 LOGN1345)  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.

Restricting 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 within each day. To control when the system may be accessed from the terminal, use the WHEN operand on the RDEFINE and RALTER commands for the TERMINAL class. For more information on the time and day-of-week controls, see "Limiting When a User Can Access the System" on page 78, or the command descriptions in *z/VM: RACF Security Server 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 logging onto those terminals to access data that has a security label higher than the terminal's security label.

Note: All of these terminals must be defined to the TERMINAL class.

Additionally, when the SECLABEL class is active and SETROPTS MLACTIVE(FAILURES) has been issued, profiles in the TERMINAL class are required to contain a SECLABEL. Therefore, logon requests for a terminal which does not have an assigned SECLABEL will fail.

Chapter 11. Protecting the z/VM Shared File System (SFS)

RACF provides security for the z/VM shared file system (SFS) by acting as the external security manager (ESM) for SFS. As documented in *z/VM: CMS File Pool Planning, Administration, and Operation,* an ESM can augment or replace the security provided with SFS on z/VM.

RACF provides the ability to protect the following items in SFS:

- SFS directories
- SFS files
- SFS external objects
- SFS administrator commands
- SFS operator commands

This chapter includes the following topics:

- Controlling access to SFS files and directories:

  General information you should know about protecting SFS files and directories.
- Implementing RACF protection for SFS files and directories:

  Steps for implementing RACF protection for SFS files and directories
- Controlling the use of SFS administrator and operator commands:

  General information and implementation steps for protecting SFS administrator and operator commands
- Setting up DMSESM PROFILE for RACF SFS protection:

  Plan which settings you need in the DMSESM PROFILE to specify how SFS will call RACF.
- Activating RACF as the SFS external security manager:

  Steps for activating the RACF protection of SFS.
- SFS administration with RACF:

  RACF administration topics associated with SFS.
- End user interaction with SFS and RACF:

  RACF SFS file and directory commands that replace SFS functions.

Controlling Access to SFS Files and Directories

You can activate the FILE and DIRECTRY classes to protect files and directories in the shared file system (SFS). Using the FILE and DIRECTRY profiles, users can provide specific protection for specific files or directories, and, through the use of generic profiles, for sets of files and directories.

Some reasons for using the FILE and DIRECTRY classes provided by RACF:

- The same RACF auditing and reporting is available for FILE and DIRECTRY profiles, as for other RACF resource profiles. For example, you can audit accesses to SFS files and directories and changes to the profiles protecting them.
- You can specify a NOTIFY user ID to be notified of failed access attempts.
- You can use generic RACF profiles to protect more than one file or directory the same way. For example, you can protect:

  - All files in a specific directory, and any of its subdirectories. This includes all existing files and any that are added in the future.
  - All files of a specific file type (SCRIPT, for example) in a specific directory. This includes all existing files of the specified type and any of that type that are added in the future.
  - All subdirectories in a specific directory, and any of their subdirectories.
- If the SETROPTS GENERICOWNER option is in effect on your system, you can control who can create profiles by specifying the user ID that appears in the profile name on the OWNER operand.
- If security classifications (such as security labels) are used on your system, they can be specified in FILE and DIRECTRY profiles.

Working with FILE and DIRECTRY Profiles

To work with SFS files and directories, you should use the RACF commands provided specifically for them:

Table 20. RACF Commands Used to Work with FILE and DIRECTRY Profiles

Activity	**FILE Profiles**	**DIRECTRY Profiles**
Defining	ADDFILE	ADDDIR
Changing	ALTFILE	ALTDIR
Listing	LFILE	LDIRECT
Granting or denying access	PERMFILE	PERMDIR
Searching	SRFILE	SRDIR
Deleting	DELFILE	DELDIR

For the authority needed to issue any of these commands, see *z/VM: RACF Security Server Command Language Reference*.

RACF SFS Command Examples

- Add a FILE profile

```
RAC ADDFILE CHECK SCRIPT POOL2:ANDREW.PAYROLL UACC(NONE) NOTIFY(ANDREW)
```

- Alter a DIRECTRY profile

```
RAC ALTDIR POOL2:ANDREW.PAYROLL SECLABEL(SECRET)
```

- Delete a FILE profile

```
RAC DELFILE PAY1994 LIST3820 SERVER2:ANDREW.
```

- List a DIRECTRY profile

```
RAC LDIRECT POOL2:ANDREW.PAY1994.TAX.DEDUCT AUTHUSER HISTORY
```

- Authorize another user to read your SFS files

```
RAC PERMFILE * * SERVER2:ANDREW.** ID(LAURIE) ACCESS(READ)
```

- Copy an access list from a FILE profile to a DIRECTRY profile

```
RAC PERMDIR SERVER2:ANDREW.** FCLASS(FILE) FROM(* * SERVER:LAURIE.**) RESET
```

- Search for DIRECTRY profiles

```
RAC SRDIR LEVEL(10) FILTER(POOL2:ANDREW.**.DEDUCT)
```

- Search for FILE profiles

```
RAC SRFILE SECLABEL(SECRET) FILTER(* * SERVER2:ANDREW.PAY*.**)
```

RACF Format for SFS Directory and File Names

Profile names for SFS objects (FILE and DIRECTRY classes) are stored in the RACF database in a different format than when they are entered using the RACF SFS commands. When using these commands, RACF follows the SFS naming conventions.

The **SFS format** of an SFS directory name is:

```
file-pool-id:userid.dir1.dir2…
```

The **SFS format** of an SFS file name is:

```
filename filetype directory-id
```

or:

```
filename filetype file-pool-id:userid.dir1.dir2…
```

To make authority checking more efficient, RACF converts the SFS file or directory name to a RACF format.

The **RACF format** of SFS directory names is:

```
file-pool-id.userid.dir1.dir2
```

The **RACF format** of SFS file names is:

```
file-pool-id.userid.dir1.dir2.filename.filetype
```

Table 21 on page 173 shows some profile name examples.

Table 21. RACF and SFS Profile Name Formats

Type	SFS Format	RACF Format
Directory	FP1:OPERATOR.DIR1.DIR2.DIR3	FP1.OPERATOR.DIR1.DIR2.DIR3
Directory	FP1:OPERATOR.	FP1.OPERATOR
File	ONE SCRIPT FP2:OPER.DIR1.DIR2	FP2.OPER.DIR1.DIR2.ONE.SCRIPT
File	MY SCRIPT FP2:OPER.	FP2.OPER.MY.SCRIPT

The SFS format of FILE and DIRECTRY profile names is used in the:

- RACF SFS file commands (ADDFILE, ALTFILE, DELFILE, LFILE, PERMFILE, and SRFILE)
- RACF SFS directory commands (ADDDIR, ALTDIR, DELDIR, LDIRECT, PERMDIR, and SRDIR)
- Output of the LDIRECT, LFILE, SRDIR, and SRFILE commands.

The RACF format of FILE and DIRECTRY profile names is:

- Used to store the profile name in the RACF database
- Used to define an entry for the FILE or DIRECTRY class in the global access checking table
- Used to process FILE and DIRECTRY profiles in RACF commands other than the RACF SFS commands (RALTER, RDEFINE, and RLIST, for example)
- Used on the RACROUTE macro when specifying a resource name (ENTITY and ENTITYX keywords, for example)
- Used in audit records
- Displayed in reports produced by such RACF utilities as DSMON, RACUT200, and the RACF report writer

- Displayed in the output of the RACF database unload utility and the SMF data unload utility.

For more information about specifying FILE and DIRECTRY profile names, see *z/VM: RACF Security Server Command Language Reference*.

Protecting SFS External Objects

RACF protects SFS external objects as resources in the FILE class. An *SFS external object* is an SFS directory entry. This entry contains a *remote name*, which is the name of an object that is not managed by the local SFS server. An SFS external object might not be an "object" at all; it is a character string stored in SFS catalogs and its use is defined by the application.

For example, if you have an SFS external object named EXTOBJ1 INFO in your FPOOL1:ANDREW.DEPT22 directory, the RACF resource name in the FILE class is:

```
EXTOBJ1 INFO FPOOL1:ANDREW.DEPT22
```

Use the RACF SFS file commands (ADDFILE, ALTFILE, DELFILE, LFILE, PERMFILE, and SRFILE) to create, update, and delete profiles for SFS external objects.

If there is no RACF profile protecting an SFS external object, access to the external object will be denied.

Access Authority for SFS Files and Directories

SFS files and directories on z/VM can have one of these access authorities:

NONE
The user or group is denied access to the SFS file or directory.

Attention:
Anyone who has READ, UPDATE, CONTROL, or ALTER authority to a protected SFS file or directory can create copies of the data in them. If a user copies the data files to an SFS file or directory for which he or she can control the security characteristics, the user can downgrade the security characteristics of the copied files. For this reason, you will probably want to assign a UACC of NONE, and then selectively permit a small number of users to access your SFS file or directory, as their needs become known. (See *z/VM: RACF Security Server General User's Guide* for information on how to permit selected users or groups to access an SFS file or directory.)

READ
The user or group is authorized to access the SFS file or directory for reading only.

UPDATE
The user or group is authorized to access the SFS file or directory for reading or writing only.

CONTROL
Equivalent to UPDATE.

ALTER
Lets users read, update, erase, discard, rename, or relocate the SFS file or directory.

When ALTER is specified in a:

- Discrete profile, users can read, alter, and delete the profile itself, *including the access list*. However, ALTER does not allow users to change the owner of the profile.
- Generic profile, users have *no* authority over the profile itself.
- Generic DIRECTRY profile, users can create SFS directories protected by the profile.
- Generic FILE profile, users can create SFS files protected by the profile.

Note: The actual access authorities required for specific SFS operations depends on the operation itself. Multiple authorities might be required.

How SFS Interacts with RACF Profiles and Authorizations

When RACF provides security for SFS, many authorities that are required for SFS functions without RACF are also required with RACF. For example, file or directory ownership is required for some commands.

If SFS requires read authority to a file or directory for a specific function, RACF READ authority is required. If SFS requires read/write authority for a function, RACF UPDATE authority is required. For example, to change or edit an existing SFS file in a FILECONTROL or DIRCONTROL directory, you must have UPDATE access to the FILE profile that protects the SFS file.

RACF profiles are not created automatically as a result of the RACROUTE interface between SFS and RACF, but profiles may be deleted or renamed. These cases are noted in the following list.

Attention:
If there is no RACF profile protecting an SFS file, directory, or external object, access to the resource will be denied.

Here is a list of common SFS functions and their corresponding required RACF authorities.

1. Read a file in a FILECONTROL or DIRCONTROL directory

   RACF authorization required:

   - READ access to the file
2. Write to an existing file in a FILECONTROL or DIRCONTROL directory

   RACF authorization required:

   - UPDATE access to the file
3. Read a directory (using LISTDIR, for example)

   RACF authorization required:

   - READ access to the directory
4. Create a directory

   RACF authorization required:

   - UPDATE access to the parent directory
   - ALTER access to the new directory name

     Note: This does not create a discrete DIRECTRY profile, but rather it checks your access to an existing DIRECTRY profile. If no profile exists to protect the new directory, the request will be denied. You must then create a DIRECTRY profile to protect the directory, and reissue the command to create the directory.
5. Create a file or external object in a FILECONTROL or DIRCONTROL directory

   RACF authorization required:

   - UPDATE access to the parent directory
   - ALTER access to the new file or external object name

     Note: This does not create a discrete FILE profile, but rather it checks your access to an existing FILE profile. If no profile exists to protect the new file or external object, the request will be denied. You must then create a FILE profile to protect the file or external object, and reissue the command to create the file or external object.
6. Create an alias in a FILECONTROL directory

   RACF authorization required:

   - READ access to the base file.
   - UPDATE access to the target directory where the alias will reside.

- The target directory's owner (if different from the command issuer) must have READ access to the base file.

Note: No RACF checking is performed on the name of the alias itself. An alias is protected by the profile which protects the base file.

7. Erase or discard a directory

   RACF authorization required:

   - UPDATE access to the parent directory
   - ALTER access to the directory being deleted

   After SFS has deleted the directory, RACF is called to also delete RACF protection from the directory. If a discrete profile is protecting the directory, the DIRECTRY profile will be deleted. If a generic profile is protecting the directory, the profile is not deleted, but an audit record may be written to record the directory deletion.

   Note: If the FILES option is used to erase or discard a directory that contains files, authorization to erase or discard each file is also required. (See the next list item.)

8. Erase or discard a file or external object in a FILECONTROL or DIRCONTROL directory

   RACF authorization required:

   - UPDATE access to the parent directory
   - ALTER access to the file or external object being deleted

   After SFS has deleted the file or external object, RACF is called to also delete RACF protection from the file or external object. If a discrete profile is protecting the file or external object, the FILE profile will be deleted. If a generic profile is protecting the file or external object, the profile is not deleted, but an audit record may be written to record the file or external object deletion.

9. Erase or discard an alias in a FILECONTROL directory

   RACF authorization required:

   - UPDATE access to the parent directory
   - READ access to the base file

10. Relocate a file or external object in a FILECONTROL directory

    RACF authorization required:

    - ALTER access to the file or external object being relocated
    - UPDATE access to the parent directory
    - UPDATE access to the target parent directory
    - Authority to create the new FILE profile (same as requirements for issuing ADDFILE) if a discrete profile exists for the old file, *or,* ALTER access to the generic profile that will protect the new file if a generic profile was protecting the old file

    After SFS has relocated the file, RACF is called to also *relocate* RACF protection for the file. If a discrete profile is protecting the file, the FILE profile will be renamed. If a generic profile is protecting the file, the profile is not deleted, but an audit record may be written to record the file being renamed.

11. Relocate an alias in a FILECONTROL directory

    RACF authorization required:

    - UPDATE access to the parent directory
    - UPDATE access to the target parent directory
    - READ access to the base file

12. Rename a file or external object in a DIRCONTROL or FILECONTROL directory

    RACF authorization required:

- ALTER access to the file or external object being renamed
- UPDATE access to the parent directory
- Authority to create the new FILE profile (same as requirements for issuing ADDFILE) if a discrete profile exists for the old file, *or,* ALTER access to the generic profile that will protect the new file if a generic profile was protecting the old file

After SFS has renamed the file, RACF is called to also *rename* RACF protection for the file. If a discrete profile is protecting the file, the FILE profile will be renamed. If a generic profile is protecting the file, the profile is not deleted, but an audit record may be written to record the file being renamed.

13. Rename an alias in a FILECONTROL directory

    RACF authorization required:

    - READ access to the base file
    - UPDATE access to the parent directory

14. Protecting a DIRCONTROL directory and its files

    RACF has no knowledge of which directories are DIRCONTROL directories; therefore, RACF treats them the same. RACF profiles in the DIRECTRY and FILE classes protect objects in a DIRCONTROL directory just as they do in a FILECONTROL directory. Common SFS functions related to DIRCONTROL and FILECONTROL directories and their RACF authorization requirements are shown above.

    RACF does not support the SFS authorities DIRREAD and DIRWRITE, but generic profiles may be used to provide the same support.

    - DIRREAD authority

      DIRREAD authority allows a user to read the directory, all files in the directory, and all files added in the future. To protect a directory in this way with RACF, you need two profiles:

      a. A DIRECTRY profile which protects the directory. For example:

      ```
      RESEARCH:USER1.DIRC
      ```

      b. A FILE profile which protects all of the files in the directory. For example:

      ```
      * * RESEARCH:USER1.DIRC
      ```

      If you gave USER2 READ access to both of these profiles, then USER2 would have the equivalent of DIRREAD access. This is true only if you do not create more specific profiles for individual files within the directory, such as:

      ```
      *       SCRIPT RESEARCH:USER1.DIRC
      PRIVATE EXEC   RESEARCH:USER1.DIRC
      ```

      Once you create these more specific FILE profiles, USER2 may no longer have the equivalent of DIRREAD authority.

    - DIRWRITE authority

      DIRWRITE authority allows a user to read from and write to the directory, all files in the directory, and all files added in the future. To protect a directory in this way with RACF, you need two profiles as listed in the examples above.

      If you gave USER2 UPDATE access to both of these profiles, then USER2 would have the equivalent of DIRWRITE access. This is true only if you do not create more specific profiles for individual files within the directory, as described for DIRREAD.

Restrictions for SFS Directory RENAME and RELOCATE

The SFS functions for renaming a directory and relocating a directory involve renaming many subdirectories and underlying files in one command. The RACROUTE interface between SFS and RACF does not support these multiple renames/relocates. When a directory is relocated or renamed, a

RACROUTE call is generated to rename the directory itself, but no calls are generated to rename underlying subdirectories and files. So if you have discrete profiles protecting files or directories which are in the structure beneath the directory being renamed or relocated, these profiles will not be renamed and will no longer protect their corresponding objects. Also, if you have generic profiles protecting only files or directories which are in the structure beneath the directory being renamed or relocated (and they protect no files and directories in the structure above it), these profiles will not be renamed and will no longer protect any objects.

To prevent this situation from occurring, the RENAME DIRECTORY and RELOCATE DIRECTORY commands should not be used when RACF is being used to protect SFS files and directories. The use of these commands can be controlled using SFSCMD profiles beginning with RENAME and RELOCATE. The RACF administrator should create profiles as follows:

```
RAC SETROPTS GENERIC(SFSCMD) CLASSACT(SFSCMD)

RAC RDEFINE SFSCMD RENAME.** UACC(NONE)
RAC PERMIT RENAME.** CLASS(SFSCMD) RESET

RAC RDEFINE SFSCMD RELOCATE.** UACC(NONE)
RAC PERMIT RELOCATE.** CLASS(SFSCMD) RESET
```

Instead of using the RENAME DIRECTORY and RELOCATE DIRECTORY commands, the ICHDIRMV EXEC may be used. See "Using the ICHDIRMV EXEC" on page 178 for more information.

If the RENAME DIRECTORY and RELOCATE DIRECTORY commands are not protected or allowed only for certain users, the following list describes the RACF authorizations required.

1. Relocate a directory

   RACF authorization required:

   - UPDATE access to the SFSCMD profile that protects the RELOCATE command
   - UPDATE access to the parent directory
   - ALTER access to the directory being relocated
   - UPDATE access to the target parent directory
   - Authority to create the new DIRECTRY profile (same as requirements for issuing ADDDIR)

   After SFS has relocated the directory, RACF is called to also *relocate* the directory. If a discrete profile is protecting the directory, the DIRECTRY profile will be renamed. If a generic profile is protecting the directory, the profile is not deleted, but an audit record may be written to record the directory being renamed.

2. Rename a directory

   RACF authorization required:

   - UPDATE access to the SFSCMD profile which protects the RENAME command
   - UPDATE access to the parent directory
   - ALTER access to the directory being renamed
   - Authority to create the new DIRECTRY profile (same as requirements for issuing ADDDIR)

   After SFS has renamed the directory, RACF is called to also *rename* the directory. If a discrete profile is protecting the directory, the DIRECTRY profile will be renamed. If a generic profile is protecting the directory, the profile is not deleted, but an audit record may be written to record the directory being renamed.

Using the ICHDIRMV EXEC

The ICHDIRMV EXEC moves or renames one of your directory structures. This EXEC provides the same function as the CMS RENAME DIRECTORY and RELOCATE DIRECTORY commands. The syntax of ICHDIRMV is:

```
ichdirmv
 dirid1 dirid2
```

where:

dirid1
Is the name of the parent directory (top node) of the directory structure that is to be moved or renamed.

dirid2
Is the name of the new parent directory.

ICHDIRMV does the following:

1. Examines the structure of subdirectories and files under *dirid1*
2. Creates a similar directory structure under *dirid2*
3. Relocates all files in *dirid1* and its subdirectories with all their files into the new structure
4. Deletes *dirid1* and its directory structure
5. Moves corresponding RACF profiles from the old directory structure to the new directory structure.

The requirements for running ICHDIRMV for your own directory are:

- You must have at least one R/W disk or directory available in the search order that is not in the directory structure being moved.
- RACF must be active as the external security manager (ESM) for SFS, providing protection for SFS files and directories.
- Top-level generic profiles should exist for the directory structure being manipulated (directory profile FP:USER1.** and file profile * * FP:USER1.**, for example).

In addition, the requirements for running ICHDIRMV to move another user's directory include being authorized to issue the commands in Table 22 on page 179 for the other user's directories:

Table 22. Authorization required to move another user's directory

Command	**RACF Authority Required**
CREATE LOCK	UPDATE authority to all subdirectories being moved
CREATE DIRECTORY	UPDATE authority to parent directories and ALTER to new directory names
RELOCATE file	If discrete RACF file profiles exist, you need group- or system-SPECIAL authority to rename them
RAC ADDDIR and RAC ADDFILE	Group- or system-SPECIAL authority
RAC DELDIR and RAC DELFILE	ALTER authority to profiles or group- or system-SPECIAL authority
DIRATTR (only issued if you are moving at least one DIRCONTROL directory)	SFSCMD authority to issue the DIRATTR command (or SFS file pool administrator authority if RACF is not protecting SFS administrator commands)
ERASE directory	ALTER authority to each subdirectory being moved

RACF Protection May Change

When ICHDIRMV is run, only RACF profiles within the directory structure being moved are moved to the new structure. There may be profiles in the old structure that protect more than just the files and directories in the old structure. *As a result, the RACF protection of some files or directories may be changed once the directory is moved to the new structure.* For example, suppose the following directories exist:

Table 23. RACF protection preceding ICHDIRMV command	
SFS directories	**RACF directory profile protecting the directory**
FP:USER1.	FP:USER1.**
FP:USER1.DIR1	FP:USER1.**
FP:USER1.DIR1.SUB1	FP:USER1.**
FP:USER1.DIR2	FP:USER1.DIR2.**

After the command `ICHDIRMV FP:USER1.DIR1 FP:USER1.DIR2` is issued to relocate the DIR1 directory, the results are:

Table 24. RACF protection following ICHDIRMV command	
SFS directories	**RACF directory profile protecting the directory**
FP:USER1.	FP:USER1.**
FP:USER1.DIR2	FP:USER1.DIR2.**
FP:USER1.DIR2.DIR1	FP:USER1.DIR2.** (was FP:USER1.**)
FP:USER1.DIR2.DIR1.SUB1	FP:USER1.DIR2.** (was FP:USER1.**)

The protection of the files within these directories may also change, depending on which file profiles existed before the directory move.

Debugging Options

The ICHDIRMV EXEC provides two keywords for debugging purposes. The syntax is:

```
ichdirmv
 dirid1 dirid2 [show [nocmd]]
```

where:

show
Displays each CMS or RACF command as it is run

nocmd
Displays each CMS or RACF command that would be run, but does not actually run the commands, so the directory move is *not* performed

Recovery Process for ICHDIRMV

As the ICHDIRMV EXEC runs, it creates a recovery or backout file called `ICHDIRMV $$BACK$$`. This backout file contains backout commands corresponding to each SFS or RACF command that completes successfully. For example, if a `CREATE LOCK` command is issued successfully, a `DELETE LOCK` command is written to the backout file.

If an error occurs during processing, the backout commands are run to restore the user's original environment. If the backout is performed successfully, the backout file is erased. If any backout command fails, the successful and failing backout commands are recorded in the backout file also. In this case, the user should correct the errors that caused the backout commands to fail and then *reissue ICHDIRMV with no parameters*. ICHDIRMV will attempt to reissue the backout commands to restore the user's original environment. Again, the backout file will be updated to reflect the current status of the backout commands.

If ICHDIRMV is interrupted by a system outage or the file pool server becomes unavailable, the backout file will contain the commands issued before the interruption occurred. When the situation is corrected, the user should reissue ICHDIRMV with no parameters, and ICHDIRMV will attempt to back out any commands that ran successfully before ICHDIRMV was interrupted.

In a case where ICHDIRMV is interrupted unexpectedly, or an error occurs when changes are being backed out, directories in the structure being moved may remain locked. When ICHDIRMV is reissued, recovery will be attempted, and the locks will be removed.

Figure 14 on page 181 shows a sample backout file. *This file should never be edited.* In this example:

- The directory names entered when ICHDIRMV was issued are shown as DIRID1 and DIRID2.
- The commands prefixed by BACKOUTCMD correspond to each CMS or RACF command issued successfully before an error or interruption occurred.
- The commands prefixed by BACKOUTGOOD are backout commands that ran successfully during backout processing.
- The commands prefixed by BACKOUTFAIL are backout commands that failed during backout processing. The failing commands should be corrected before running ICHDIRMV again.

```
**PLEASE MAKE NO CHANGES TO THIS FILE**
DIRID1: RESEARCH:USER1.CDC1
DIRID2: RESEARCH:USER1.CDC1NEW
BACKOUTCMD: DELETE LOCK RESEARCH:USER1.CDC1
BACKOUTCMD: DELETE LOCK RESEARCH:USER1.CDC1.SUB1
BACKOUTCMD: DELETE LOCK RESEARCH:USER1.CDC1.SUB2
BACKOUTCMD: RAC DELDIR RESEARCH:USER1.CDC1NEW.**
BACKOUTCMD: RAC DELFILE * * RESEARCH:USER1.CDC1NEW.**
BACKOUTCMD: ERASE RESEARCH:USER1.CDC1NEW
BACKOUTCMD: ERASE RESEARCH:USER1.CDC1NEW.SUB1
BACKOUTCMD: ERASE RESEARCH:USER1.CDC1NEW.SUB2
BACKOUTCMD: DIRATTR RESEARCH:USER1.CDC1 DIRCONTROL
BACKOUTCMD: DIRATTR RESEARCH:USER1.CDC1.SUB1 DIRCONTROL
BACKOUTCMD: DIRATTR RESEARCH:USER1.CDC1.SUB2 DIRCONTROL
BACKOUTCMD: RELOCATE * * RESEARCH:USER1.CDC1NEW TO RESEARCH:USER1.CDC1
BACKOUTCMD: RELOCATE * * RESEARCH:USER1.CDC1NEW.SUB1 TO RESEARCH:USER1.CDC1.SUB1
BACKOUTCMD: RELOCATE * * RESEARCH:USER1.CDC1NEW.SUB2 TO RESEARCH:USER1.CDC1.SUB2
*=== END of Commands ===*
*=== BACKOUT starting ===*
BACKOUTGOOD: RELOCATE * * RESEARCH:USER1.CDC1NEW.SUB2 TO RESEARCH:USER1.CDC1.SUB2
BACKOUTGOOD: RELOCATE * * RESEARCH:USER1.CDC1NEW.SUB1 TO RESEARCH:USER1.CDC1.SUB1
BACKOUTGOOD: RELOCATE * * RESEARCH:USER1.CDC1NEW TO RESEARCH:USER1.CDC1
BACKOUTFAIL: DIRATTR RESEARCH:USER1.CDC1.SUB2 DIRCONTROL
BACKOUTFAIL: DIRATTR RESEARCH:USER1.CDC1.SUB1 DIRCONTROL
BACKOUTFAIL: DIRATTR RESEARCH:USER1.CDC1 DIRCONTROL
BACKOUTGOOD: ERASE RESEARCH:USER1.CDC1NEW.SUB2
BACKOUTGOOD: ERASE RESEARCH:USER1.CDC1NEW.SUB1
BACKOUTGOOD: ERASE RESEARCH:USER1.CDC1NEW
BACKOUTGOOD: RAC DELFILE * * RESEARCH:USER1.CDC1NEW.**
BACKOUTGOOD: RAC DELDIR RESEARCH:USER1.CDC1NEW.**
BACKOUTGOOD: DELETE LOCK RESEARCH:USER1.CDC1.SUB2
BACKOUTGOOD: DELETE LOCK RESEARCH:USER1.CDC1.SUB1
BACKOUTGOOD: DELETE LOCK RESEARCH:USER1.CDC1
```

Figure 14. Sample ICHDIRMV $$BACK$$ File

The SFSAUTOACCESS Option

The SFSAUTOACCESS option, a performance enhancement for SFS FILE and DIRECTRY protection in RACF, reduces the number of RACROUTE calls made from SFS to RACF. This option allows the RACROUTE REQUEST=AUTH interface running in the SFS file pool server to automatically grant access to a file or directory if a user is accessing his or her own SFS file or directory. When automatically granting access, the call to the RACF service machine is bypassed, resulting in substantial performance improvements.

For more information, see *z/VM: RACF Security Server System Programmer's Guide*.

Security Label Considerations for SFS Files and Directories

When the SECLABEL class is active and a security label exists in the FILE or DIRECTRY profile protecting the resource, RACF performs SECLABEL authorization checking. The type of SECLABEL authorization required is based on the type of access being requested for the SFS file or directory.

If the user requests an operation that requires:

1. READ access to the FILE or DIRECTRY profile, a read-only authorization request is performed.

2. READ, UPDATE, CONTROL, or ALTER access to the FILE or DIRECTRY profile, a read/write authorization request is performed.

The outcome of the request depends on the SETROPTS options that are in effect. For more information, see “Security Label Authorization Checking” on page 313.

When the SECLABEL class is active and SETROPTS MLACTIVE(FAILURES) has been issued, profiles in the FILE and DIRECTRY classes must contain a SECLABEL. Authorization requests for SFS files and directories without SECLABELs assigned to them will fail.

Implementing RACF Protection for SFS Files and Directories

To protect SFS files and directories using RACF, you'll need to:

1. Migrate existing SFS authorities into RACF using ICHSFS
2. Plan your changes to the DMSESM PROFILE
3. Activate RACF as the SFS external security manager

Step 1: Migrate Existing SFS Authorities Into RACF Using ICHSFS

You can use the ICHSFS EXEC to migrate existing SFS authorities into RACF. This EXEC does the following:

- Extracts SFS access control information for user-specified file pools by file and directory.
- For each file pool, generates a batch file of appropriate RACF commands to add equivalent access control information to the RACF database. The batch file has a file name equal to the file pool being migrated, and a file type of SFSUT1.
- When requested by the user, executes the batch file of RACF commands.
- If files with file type SFSUT1 exist when ICHSFS is invoked, the EXEC asks whether the files should be executed or deleted. If a user chooses not to execute or delete a batch file for a file pool, and later specifies that same file pool to be migrated, the existing batch file is erased and recreated.

Note: RACF does not recognize lowercase characters in profile names. The ICHSFS EXEC places any file names with lowercase characters in a file. You can use generic characters (including %) to protect these files. For example, to protect the file:

```
OFSMAIL OFSLOGfl POOL1:USER1.DIR1
```

you could use any of these file profile names:

```
OFSMAIL OFSLOG*  POOL1:USER1.DIR1
OFSMAIL OFSLOG%% POOL1:USER1.DIR1
*       OFSLOG%% POOL1:USER1.DIR1
*       OFSLOG%% POOL1:USER1.**
```

The ICHSFS EXEC can be used in two ways:

- Interactively, to process one file pool at a time
- Automatically, to process file pools listed in an input file.

Using ICHSFS Interactively

To use ICHSFS interactively, do the following:

1. While in a CMS ready state, enter:

   ```
   ichsfs
   ```

   In response, the EXEC prompts for the file pool to be migrated.
2. Enter the file pool ID.

The EXEC creates a batch file of RACF commands for the file pool specified, and asks whether the batch file should be executed.

3. Enter yes or no.

   If the response is yes, the EXEC executes the batch file and prompts for another file pool to be migrated. If the response is no, the EXEC saves the batch file on the first R/W disk or directory in your search order.

Using ICHSFS Automatically

To use ICHSFS automatically, do the following:

1. Create a CMS file that lists the IDs of the file pools to be migrated. List one ID per line in the file, starting in column 1.
2. While in a CMS ready state, enter the following command:

   ```
   ichsfs filename filetype [filemode]
   ```

   where:

   filename
   is the file name of the file you created.

   filetype
   is the file type of the file you created.

   filemode
   is the file mode of the file you created. The file mode is optional. If it is not specified, it defaults to the first R/W disk or directory in your search order.

   ICHSFS processes each file pool ID listed in the input file, creates a batch file of RACF commands for each one, and places the batch files on the first R/W disk or directory in your search order. When all of the file pools have been processed, the EXEC asks, one at a time, whether each batch file should be executed.
3. You can choose to execute each batch file, all of them, or none of them.

How SFS Authorities are Translated into RACF Authorities

To give all enrolled SFS users full access to their own SFS files and directories, the following steps are taken in the ICHSFS utility.

1. For every enrolled user, the following commands are issued:

   ```
   ADDDIR dirid.** OWNER(userid) UACC(NONE)
   PERMDIR dirid.** RESET
   ADDFILE * * dirid.** OWNER(userid) UACC(NONE)
   PERMFILE * * dirid.** RESET
   ```

2. For every directory, the following commands are issued:

   ```
   ADDDIR dirid OWNER(userid) UACC(NONE)
   PERMDIR dirid RESET
   ADDFILE * * dirid OWNER(userid) UACC(NONE)
   PERMFILE * * dirid RESET
   ```

3. For each file in a file control directory which has existing authorities granted to other users, the following commands are issued:

   ```
   ADDFILE fn ft dirid OWNER(userid) UACC(NONE)
   PERMFILE fn ft dirid RESET
   ```

Also, the file pool administrator who is running the ICHSFS utility is *not* given any authorities to the files and directories being migrated.

Table 25 on page 184 describes how previously issued GRANT commands (except those for <PUBLIC>) are translated into equivalent RACF authorities. Where there is not a direct equivalent in RACF for an SFS authority, the table notes the differences.

Table 25. SFS Authorities Translated As RACF Authorities

Type	SFS Authority	Corresponding RACF Authority
File control directory	READ	PERMDIR dirid ID(grantee) ACCESS(READ)
File control directory	WRITE	PERMDIR dirid ID(grantee) ACCESS(UPDATE) PERMFILE * * dirid ID(grantee) ACCESS(ALTER) **Note:** This also gives the equivalent of NEWWRITE authority.
File control directory	NEWREAD	PERMFILE * * dirid ID(grantee) ACCESS(READ)
File control directory	NEWWRITE	PERMFILE * * dirid ID(grantee) ACCESS(ALTER)
File in a File control directory	READ	PERMFILE fn ft dirid ID(grantee) ACCESS(READ)
File in a File control directory	WRITE	PERMFILE fn ft dirid ID(grantee) ACCESS(UPDATE) **Note:** The grantee is not allowed to erase the file.
Directory control directory	DIRWRITE	PERMDIR dirid ID(grantee) ACCESS(UPDATE) PERMFILE * * dirid ID(grantee) ACCESS(ALTER)
Directory control directory	DIRREAD	PERMDIR dirid ID(grantee) ACCESS(READ) PERMFILE * * dirid ID(grantee) ACCESS(READ)

Table 26 on page 184 describes how previously issued GRANT commands for <PUBLIC> are translated into equivalent RACF authorities. Where there is not a direct equivalent in RACF for an SFS authority, the table notes the differences.

Table 26. SFS <PUBLIC> Authorities Translated As RACF Authorities

Type	SFS <PUBLIC> Authority	Corresponding RACF authority
File control directory	READ	ALTDIR dirid UACC(READ)
File control directory	WRITE	ALTDIR dirid UACC(UPDATE) ALTFILE * * dirid UACC(ALTER) **Note:** This also gives the equivalent of NEWWRITE authority.
File control directory	NEWREAD	ALTFILE * * dirid UACC(READ)
File control directory	NEWWRITE	ALTFILE * * dirid UACC(ALTER)
File in a File control directory	READ	ALTFILE fn ft dirid UACC(READ)
File in a File control directory	WRITE	ALTFILE fn ft dirid UACC(UPDATE) **Note:** Users are not allowed to erase the file.
Directory control directory	DIRWRITE	ALTDIR dirid UACC(UPDATE) ALTFILE * * dirid UACC(ALTER)

Table 26. SFS <PUBLIC> Authorities Translated As RACF Authorities (continued)		
Type	**SFS <PUBLIC> Authority**	**Corresponding RACF authority**
Directory control directory	DIRREAD	ALTDIR dirid UACC(READ) ALTFILE * * dirid UACC(READ)

Step 2: Plan Your Changes to the DMSESM PROFILE

The RACF protection of SFS files and directories is dependent on ESM calls from an SFS file pool server, and the DMSESM PROFILE defines which calls are made to RACF. The parameters of concern for SFS file and directory protection are the A*nnnn* parameter and the last two characters of the B*nnnn* parameter in Record 2.

See "Setting Up DMSESM PROFILE for RACF SFS Protection" on page 187 for more information.

Step 3: Activate RACF as the SFS External Security Manager

If you are activating RACF protection for the first time on your file pool server, follow the instructions in "Activating RACF as the SFS External Security Manager" on page 189.

If you are already using RACF to protect SFS administrator and operator commands or BFS files in your file pool, you must update the DMSESM PROFILE with the changes you planned in step 2, and restart your file pool server.

Controlling the Use of SFS Administrator and Operator Commands

You can activate the SFSCMD class to control who can issue SFS administrator and operator commands. Using SFSCMD profiles, users can provide protection for specific commands, and, through the use of generic profiles, for sets of commands.

The SFSCMD class is also used to restrict the use of the general user commands for RENAME DIRECTORY and RELOCATE DIRECTORY. See "Restrictions for SFS Directory RENAME and RELOCATE" on page 177 for more information.

When RACF is called to authorize an SFS administrator command, this replaces the SFS checking for file pool administrator authority, as specified in the DMSPARMS file or with the GRANT ADMIN command. RACF will authorize or deny use of the command by checking the command issuer's access to the SFSCMD profile protecting the command.

When RACF is protecting SFS operator commands, you must still enter the commands from the file pool server's console or from its secondary user console. Access to issue the commands can be further limited using SFSCMD profiles. RACF will authorize or deny use of the operator command by checking the command issuer's access to the SFSCMD profile protecting the command.

When SFS calls RACF to authorize an administrator or operator command and there is no profile to protect that command, RACF will fail the command.

To control which users can issue SFS administrator and operator commands, take the following steps:

1. Plan your changes to the DMSESM profile.

   The RACF protection of SFS administrator and SFS operator commands is dependent on ESM calls from an SFS file pool server using the RACROUTE interface. Using the DMSESM PROFILE as described in "Setting Up DMSESM PROFILE for RACF SFS Protection" on page 187, an installation can specify that RACF be called for all, some, or none of the SFS administrator and operator commands. The parameters of concern in the DMSESM PROFILE are the B*nnnn* and C*nn* parameters in Record 2.
2. If you plan to use generic profiles in the SFSCMD class, activate generic profiles:

   ```
   SETROPTS GENERIC(SFSCMD)
   ```

IBM also recommends that you create a "top" generic profile in the SFSCMD class to protect those SFS administrator and operator commands that are not protected by a more specific profile. To do so, enter:

```
RDEFINE SFSCMD ** UACC(NONE)
```

3. Create appropriate profiles in the SFSCMD class:

   ```
   RDEFINE  SFSCMD  profile-name  UACC(NONE)
   ```

   Specifying UACC(NONE) is recommended if most users on the system will *not* be able to issue SFS administrator and operator commands. The resource names are determined by z/VM, and are documented in *z/VM CMS File Pool Planning, Administration, and Operation.*

   For example, the SFS administrator command

   ```
   QUERY FILEPOOL REPORT (CATALOG
   ```

   is translated into the resource name

   ```
   filepool.QUERY.FILEPOOL.CATALOG
   ```

   in the SFSCMD class. This resource name could be protected by a RACF profile in the SFSCMD class named

   ```
   *.QUERY.FILEPOOL.CATALOG
   ```

   For example, the SFS operator command

   ```
   AUDIT ON ALL
   ```

   is translated into the resource name

   ```
   filepool.AUDIT.ON.ALL
   ```

   in the SFSCMD class. This resource name could be protected by a RACF profile in the SFSCMD class named

   ```
   *.AUDIT.ON.ALL
   ```

4. Use the PERMIT command to allow appropriate users and groups access to the profile by giving them at least UPDATE access:

   ```
   PERMIT profile-name CLASS(SFSCMD)  ID(userid|group)
          ACCESS(UPDATE)
   ```

   For example:

   ```
   PERMIT *.AUDIT.ON.ALL CLASS(SFSCMD) ID(LAURIE) ACCESS(UPDATE)
   ```

5. When you are ready to use the protection defined in SFSCMD profiles, activate the SFSCMD class:

   ```
   SETROPTS CLASSACT(SFSCMD)
   ```

6. For performance reasons, you should consider requesting SETROPTS RACLIST processing for the SFSCMD class:

   ```
   SETROPTS RACLIST(SFSCMD)
   ```

7. Activate RACF as the SFS external security manager.

   If you are activating RACF protection for the first time on your file pool server, follow the instructions in “Activating RACF as the SFS External Security Manager” on page 189.

If you are already using RACF to protect SFS files and directories or BFS files in your file pool, you must update the DMSESM PROFILE with the changes you planned in step 1 above, and restart your file pool server.

Security Label Considerations for the SFSCMD class

When the SECLABEL class is active and a security label exists in the SFSCMD profile protecting the command, RACF performs SECLABEL authorization checking. A read/write authorization request is performed, and the outcome of the request depends on the SETROPTS options that are in effect. For more information, see "Security Label Authorization Checking" on page 313.

Setting Up DMSESM PROFILE for RACF SFS Protection

The file DMSESM PROFILE tells the SFS file pool server which types of authorization checking will be routed to an external security manager (ESM). This section describes what this file should look like when RACF is the designated ESM for SFS.

Details about each parameter specified in DMSESM PROFILE can be found in *z/VM CMS File Pool Planning, Administration, and Operation.*

The file ICHSFSPF DMSESM is shipped with RACF. It is a sample file for DMSESM PROFILE. Its contents determine which SFS authorization calls will be routed to RACF. It is illustrated in Figure 15 on page 187. It may be modified as described below based on your installation's needs.

```
P1 DMSSECIT
A1012 DMSUAUTH B1012 DMSAAUTH C11 DMSOAUTH D1 DMSSECIT E1 DMSPERM
A002301 A002401 A002701 A002801
```

Figure 15. RACF-Supplied ICHSFSPF DMSESM

Record 1: Initialization and Termination Routine

The first line of the DMSESM PROFILE contains the name of the ESM initialization and termination routine. The line supplied with RACF specifies:

```
P1 DMSSECIT
```

where:

P1
Specifies that if RACF defers a request back to SFS (return code 4 from a RACROUTE REQUEST=AUTH, for example), REQUEST=AUTH, for example), SFS treats it as a rejected authorization. This value must not be changed.

DMSSECIT
Is the name of the routine called to initialize and terminate RACF processing for SFS. It must not be changed.

Record 2: Types of Calls to Be Reviewed

The second line of the DMSESM PROFILE identifies the types of authorization calls SFS should call RACF for. Five types of calls can be directed to RACF:

- SFS object authorization check
- SFS command authorization check
- SFS operator command authorization check
- ESM program check
- BFS object security

The line supplied with RACF specifies:

```
A1012 DMSUAUTH B1012 DMSAAUTH C11 DMSOAUTH D1 DMSSECIT E1 DMSPERM
```

where:

A1012
Specifies that RACF:

1. Checks new file, external object, and directory names as they are created
2. Does not check alias names as they are created, changed, or deleted
3. Is called by SFS for all object authorization checking
4. Is called by SFS at commit and rollback time only for specific file pool requests, identified by tokens in the third and following lines of the profile

To use SFSCMD protection with no file and directory protection, change this value to A0000. If you do this, you must also remove the A*nnnnnn* tokens on Record 3.

DMSUAUTH
Is the name of the routine called for object authorization checks. It must not be changed.

B1012
Specifies that:

1. RACF is called for SFS command authorization
2. RACF is not called at commit and rollback time for SFS command authorization
3. Renaming an SFS directory requires SFS command authorization
4. Relocating an SFS directory requires SFS command authorization

This value may be changed to B0012. This is the same as B1012 except RACF is not called for SFS command authorization. This would be used if you did not want to use SFSCMD profiles to protect any SFS administrator commands.

This value may also be changed to B2012 to specify that SFS calls RACF only for specific file pool requests, identified by tokens in the third and following lines of the profile. For example, if the third line contained B003210, SFS would call RACF for SFSCMD authorization when a delete storage request (file pool request code X'32') was made. See *z/VM CMS File Pool Planning, Administration, and Operation* for more information on using this value.

If you specify token A0000 to use SFSCMD protection with no file and directory protection, you can change this token to B1000 or B2000.

DMSAAUTH
Is the name of the routine called for command authorization checks. It must not be changed.

C11
Specifies that:

1. RACF is called for SFS operator command authorization
2. SFS will accept RACF denial of authorization for an operator command

This value may be changed to C01. This is the same as C11 except RACF is not called for SFS operator command authorization. This would be used if you did not want to use SFSCMD profiles to protect any SFS operator commands.

This value may also be changed to C21 to specify that SFS calls RACF only for specific file pool requests, identified by tokens in the third and following lines of the profile. For example, if the third line contained C000910, SFS would call RACF for SFSCMD authorization when a STOP command (special file pool request code X'09') was issued. See *z/VM CMS File Pool Planning, Administration, and Operation* for more information on using this value.

DMSOAUTH
Is the name of the routine called for operator command authorization checks. It must not be changed.

D1
Specifies that SFS should call the specified routine if a program check occurs in RACF code. It must not be changed.

DMSSECIT
Is the name of the routine called to handle a program check in RACF code. It must not be changed.

E1
Specifies that RACF should be called for permission checking for BFS objects. If RACF is not being used to protect objects in BFS, this may be changed to E0.

See "Setting up Support for OpenExtensions" on page 195 for more information.

DMSPERM
Is the name of the routine called to perform permission checking for BFS objects. It must not be changed.

Record 3: Specific File Pool Requests To Be Reviewed

The third and following lines of the DMSESM PROFILE contain tokens identifying specific file pool requests that RACF should review. Each token represents a specific file pool request on a specific type of authorization call. The line supplied with RACF specifies:

```
A002301 A002401 A002701 A002801
```

where:

A002301
Specifies that RACF is called by SFS for an object authorization check for a *delete request* call at commit and rollback.

A002401
Specifies that RACF is called by SFS for an object authorization check for a *delete directory* call at commit and rollback.

A002701
Specifies that RACF is called by SFS for an object authorization check for a *relocate* call at commit and rollback.

A002801
Specifies that RACF is called by SFS for an object authorization check for a *rename* call at commit and rollback.

These four values must not be changed. However, if A0000 is specified in Record 2, you can delete these four values. In addition, you can add more B*nnnnnn* and C*nnnnnn* tokens to Record 3.

Activating RACF as the SFS External Security Manager

1. Authorize the file pool server to use the RACROUTE interface.

   Because SFS calls RACF using the RACROUTE interface, each file pool server which will be calling RACF must be authorized to use the RACROUTE interface. See *z/VM: Security Server RACROUTE Macro Reference* for instructions on doing this authorization using the ICHCONN profile in the FACILITY class. The access required for the ICHCONN profile is UPDATE authority.

   The file pool server also must have access to the RACROUTE interface code, which is typically placed on the Y-disk (MAINT's 19E) during RACF installation. The main module for the RACROUTE interface is the RPIUCMS MODULE.

   When RACF is specified as the external security manager for SFS, the number of calls to the RACF service machine increases dramatically. For this reason, IBM recommends that at least one RACF service machine be dedicated to process RACROUTE requests from the SFS service machine. If you have more than one SFS server, you can have them all send RACROUTE requests to one RACF service machine, or assign groups of them to different RACF service machines. This can be done using multiple RACF service machines, as described in *z/VM: RACF Security Server System Programmer's Guide*.

2. Consider using the SFSAUTOACCESS option.

   If you are using RACF to protect SFS files and directories, and you want to use the SFSAUTOACCESS option, place a copy of the RACF SERVMACH file specifying the appropriate parameters on the file pool server's A-disk.

   See "The SFSAUTOACCESS Option" on page 181 and *z/VM: RACF Security Server System Programmer's Guide* for more information.

3. Specify ESECURITY as a start-up parameter in the DMSPARMS file. This file usually resides on the A-disk of each SFS file pool server and is named *server-id* DMSPARMS, where *server-id* is the user ID of the file pool server. See *z/VM CMS File Pool Planning, Administration, and Operation* for more information.
4. Put DMSESM PROFILE on the file pool server's A-disk.

   You should have already planned the contents of the DMSESM PROFILE that is appropriate for your installation. Place this copy on the file pool server's A-disk.

5. Restart your file pool server.

   When you restart your file pool server, the RACROUTE interface will be initialized. The following messages are typically issued to the file pool server's console if initialization was properly completed:

```
DMS5SB2029I Initialization begins for external security
DMS5SB2029I Initialization begins for external security routine DMSSECIT
RPICMS016I USER/RACF z/VM Racroute communication path is established.
DMS5SB2029I Initialization ends for external security routine DMSSECIT
```

SFS Administration with RACF

This section will discuss some RACF administration topics associated with SFS.

Using SPECIAL and OPERATIONS Authorities

When SFS is active without using an external security manager, an SFS administrator has full access over all resources in a file pool. For example, the administrator can read from or write to any user file in the file pool and also grant and revoke authorities to users' files and directories. With RACF, there is no direct equivalent to this type of authority for a single file pool. However, the RACF SPECIAL and OPERATIONS attributes offer similar capabilities.

The RACF SPECIAL attribute allows you to update any profile in the RACF database. This includes FILE and DIRECTRY profiles, but also all other profiles. With the SPECIAL attribute, you can update the access list in any profile to authorize users to access resources. You can also update other fields in RACF profiles such as UACC, NOTIFY, and LEVEL. For more information on the RACF SPECIAL attribute, see "SPECIAL Attribute" on page 57.

You can also use group-SPECIAL authority to limit the scope of SPECIAL authority. For more information, see "Scope of Authority for group-SPECIAL, group-AUDITOR, and group-OPERATIONS Users" on page 61.

Note: To create a FILE or DIRECTRY profile containing a second qualifier that is different from your own user ID, you must have either the SPECIAL attribute or group-SPECIAL authority.

The RACF OPERATIONS authority allows you to read and write from *any* SFS file or directory in *any* file pool. With the OPERATIONS attribute, you can also access resources in other classes, such as minidisks. For more information on the RACF OPERATIONS attribute, see "OPERATIONS Attribute" on page 59.

You can also use group-OPERATIONS authority to limit the scope of OPERATIONS authority. For more information, see "Scope of Authority for group-SPECIAL, group-AUDITOR, and group-OPERATIONS Users" on page 61.

Disallowed Commands

Table 27 on page 191 describes commands that are not allowed when SFS is running with RACF protection active. Along with each item is the RACF replacement for it and where to find more information.

Table 27. Disallowed Commands

Disallowed Commands:	Replaced With RACF:	Where To Find More Information:
End-user commands: GRANT REVOKE	RAC PERMDIR command RAC PERMFILE command	See "End-User Interaction with SFS and RACF" on page 191.
SFS administrator commands: ENROLL ADMINISTRATOR DELETE ADMINISTRATOR	SFSCMD authorities RACF SPECIAL authority RACF OPERATIONS authority	See "Controlling the Use of SFS Administrator and Operator Commands" on page 185 and "Using SPECIAL and OPERATIONS Authorities" on page 190.
SFS operator commands: GRANT ADMIN REVOKE ADMIN	SFSCMD authorities RACF SPECIAL authority RACF OPERATIONS authority	See "Controlling the Use of SFS Administrator and Operator Commands" on page 185 and "Using SPECIAL and OPERATIONS Authorities" on page 190.

Adding an SFS User

When you enroll a new user into an SFS file pool, you should also create "top" generic profiles and make the user the owner of the profiles. For example, assuming that user JOSE's file pool ID is POOL1, you would do the following:

```
RAC ADDFILE * * POOL1:JOSE.** OWNER(JOSE) UACC(NONE)
RAC ADDDIR POOL1:JOSE.** OWNER(JOSE) UACC(NONE)
```

Note:

1. All users can create and modify their own FILE and DIRECTRY profiles. Class authority, or CLAUTH, does not apply to the FILE and DIRECTRY classes.
2. Users always have access to their own files and directories. This is determined by the user ID qualifier of the FILE or DIRECTRY profile name.

End-User Interaction with SFS and RACF

This section discusses the RACF commands that replace the functions of several CMS commands.

Replacements for the GRANT and REVOKE Commands

When RACF protection is in effect for SFS, you cannot use the CMS GRANT and REVOKE commands, Instead, use the RACF PERMDIR and PERMFILE commands to authorize another user to access your SFS files and directories. Table 28 on page 191 describes how SFS authorities used on the GRANT command can be matched to equivalent RACF authorities. Where there is not a direct equivalent in RACF for an SFS authority, the table notes the differences.

Table 28. SFS Authorities Translated As RACF Authorities: For End Users

Type of Resource	SFS Authority	Corresponding RACF authority
File control directory	READ	RAC PERMDIR dirid ID(grantee) ACCESS(READ)
File control directory	WRITE	RAC PERMDIR dirid ID(grantee) ACCESS(UPDATE) RAC PERMFILE * * dirid ID(grantee) ACCESS(ALTER) **Note:** This also gives the equivalent of NEWWRITE authority.
File control directory	NEWREAD	RAC PERMFILE * * dirid ID(grantee) ACCESS(READ)
File control directory	NEWWRITE	RAC PERMFILE * * dirid ID(grantee) ACCESS(ALTER)

Table 28. SFS Authorities Translated As RACF Authorities: For End Users (continued)		
Type of Resource	**SFS Authority**	**Corresponding RACF authority**
File in a File control directory	READ	RAC PERMFILE fn ft dirid ID(grantee) ACCESS(READ)
File in a File control directory	WRITE	RAC PERMFILE fn ft dirid ID(grantee) ACCESS(UPDATE) **Note:** The grantee is not allowed to erase the file.
Directory control directory	DIRWRITE	RAC PERMDIR dirid ID(grantee) ACCESS(UPDATE) RAC PERMFILE * * dirid ID(grantee) ACCESS(ALTER)
Directory control directory	DIRREAD	RAC PERMDIR dirid ID(grantee) ACCESS(READ) RAC PERMFILE * * dirid ID(grantee) ACCESS(READ)

Table 29 on page 192 describes how you can do the equivalent of granting <PUBLIC> authority with the RACF UACC keyword. Where there is not a direct equivalent in RACF for an SFS authority, the table notes the differences.

Table 29. SFS <PUBLIC> Authorities Translated As RACF Authorities: For End Users		
Type of Resource	**SFS <PUBLIC> Authority**	**Corresponding RACF authority**
File control directory	READ	RAC ALTDIR dirid UACC(READ)
File control directory	WRITE	RAC ALTDIR dirid UACC(UPDATE) RAC ALTFILE * * dirid UACC(ALTER) **Note:** This also gives the equivalent of NEWWRITE authority.
File control directory	NEWREAD	RAC ALTFILE * * dirid UACC(READ)
File control directory	NEWWRITE	RAC ALTFILE * * dirid UACC(ALTER)
File in a File control directory	READ	RAC ALTFILE fn ft dirid UACC(READ)
File in a File control directory	WRITE	RAC ALTFILE fn ft dirid UACC(UPDATE) **Note:** Users are not allowed to erase the file.
Directory control directory	DIRWRITE	RAC ALTDIR dirid UACC(UPDATE) RAC ALTFILE * * dirid UACC(ALTER)
Directory control directory	DIRREAD	RAC ALTDIR dirid UACC(READ) RAC ALTFILE * * dirid UACC(READ)

Replacement for the QUERY AUTHORITY Command

The CMS QUERY AUTHORITY command displays the authorities for an SFS file or directory. When RACF is protecting SFS files and directories, the QUERY AUTHORITY command will display a "P" to indicate a file or directory is protected by RACF. Instead of the QUERY AUTHORITY command, use the RACF LFILE and LDIRECT commands to display the authorities for SFS files and directories.

The LFILE and LDIRECT commands list the RACF profile protecting a resource. For example, to list the profile and authorities for ANDREW's top directory in file pool POOL1, issue:

```
RAC LDIRECT POOL1:ANDREW. AUTH
```

See *z/VM: RACF Security Server General User's Guide* and *z/VM: RACF Security Server Command Language Reference* for details on using the LFILE and LDIRECT commands.

Replacement for the RENAME and RELOCATE DIRECTORY Commands

The RENAME DIRECTORY and RELOCATE DIRECTORY commands should not be used when RACF is protecting SFS files and directories. Users should use the ICHDIRMV EXEC instead. See "Restrictions for SFS Directory RENAME and RELOCATE" on page 177 and "Using the ICHDIRMV EXEC" on page 178 for more information.

Chapter 12. Controlling OpenExtensions and BFS Security

z/VM with RACF provides security at the Portable Operating System Interface (POSIX) 1003.1 level.

POSIX, a standard introduced by the Institute of Electrical and Electronics Engineers (IEEE), defines a standard set of interfaces through which a program can request services from the operating system it is running on. A *POSIX-compliant system* adheres to the specifications documented for each function defined, regardless of the differences in underlying implementations. When application developers design programs that use only the system interfaces defined by POSIX, these programs can be ported easily to other operating systems and programs that are POSIX-compliant. OpenExtensions is z/VM's implementation of POSIX.

OpenExtensions security information is stored and maintained in the RACF database in OVM segments of USER and GROUP profiles. You can work with the information in the OVM segment using the ADDUSER, ALTUSER, LISTUSER, ADDGROUP, ALTGROUP and LISTGRP commands.

VM retrieves a user's OpenExtensions information from RACF when that user logs on to the system. VM can also request OpenExtensions information from the RACF database at any time. OpenExtensions z/VM services use this OpenExtensions information for the duration of the user's logon session.

RACF performs access checking for the byte file system (BFS) on z/VM. RACF and the BFS server perform the standard POSIX access checks, based on permission bits associated with a file in the BFS. RACF also audits file and directory access checks, as well as some other OpenExtensions events, based on LOGOPTIONS and AUDIT settings for several classes in the class descriptor table. These classes, which control the auditing characteristics in a POSIX environment, are: DIRSRCH, DIRACC, FSSEC, FSOBJ, and PROCESS.

Setting up Support for OpenExtensions

To set up RACF's support of OpenExtensions, perform these steps:

1. Define OpenExtensions information in your RACF USER profiles

   See “Defining OpenExtensions Users” on page 198 for more information.
2. Define GIDs for your RACF GROUP profiles.

   See “Defining OpenExtensions Groups” on page 201 for more information.
3. Identify your BFS service machine user IDs to RACF:
   a. Create a profile called POSIXOPT.SETIDS in the VMPOSIX class.
   b. Permit the BFS service machine user IDs.
   c. If it is not active already, activate the VMPOSIX class.

      Notes:

      The BFS service machine must be logged off and then logged back on order for the change to take effect.

      If you do *not* perform this step, the BFS service machine will incur an 0C6 abend when it invokes the RACF security exit. The following messages are part of the output that will be generated as a result of this abend:

```
DMS5SB2029I External security routine DMSSECIT called due to
             program check
RPICMS017I USER/RACF z/VM Racroute communication path has been
           terminated.
DMS5SB2029I External security routine DMSSECIT program check
             processing complete
```

See “OpenExtensions BFS File Security” on page 203 for more information about the BFS.

If you already have POSIXOPT SETIDS statements defined in the CP directory, you can run the RPIDIRCT EXEC to create the RACF commands that will define this profile and do the appropriate PERMITs. See “Processing OpenExtensions Statements in the CP Directory” on page 49 for more information.

4. Decide if you want RACF to perform further authorization for executing set-UID and set-GID files in the BFS.

   See “Protecting Set-UID and Set-GID Executable Files” on page 205 for details on protecting set-UID and set-GID files.

   To use the RACF protection, perform these steps:

   a. Create the appropriate profiles.
   b. Permit users as appropriate.
   c. If it is not active already, activate the VMPOSIX class.

   By default, z/VM calls RACF for this authorization. RACF will fail the request if any of the following are true:

   - The VMPOSIX class is not active
   - The EXEC.U*uid* profile is not defined
   - The EXEC.G*gid* profile is not defined.

   Thus, by default, execution of set-id files will fail. If you do not want RACF to enforce the extra protection, turn off control for the DIAG280 event in your currently active VMXEVENT profile.

5. Determine which user IDs should be allowed to request OpenExtensions information about:

   - Groups the user is not connected to
   - Other users.

   The user information which can be retrieved consists only of the OpenExtensions related data: the information from the user's OVM segment, and the user's supplementary group list. The group information which can be retrieved consists of a list of user IDs which are connected to the group. Field level access checking is not enforced by RACF when this information is requested.

   On a POSIX system, this information is generally available to anyone. However, because users can't generally see this information using RACF commands, an optional mechanism is provided to restrict this ability.

   By default, all OpenExtensions queries are allowed. To restrict this ability:

   a. Create a profile called POSIXOPT.QUERYDB in the VMPOSIX class.
   b. Permit only those users who need to make these queries.
   c. If it is not active already, activate the VMPOSIX class.

   It is not necessary to permit superusers to this profile.

   In general, the OpenExtensions shell performs these queries on behalf of shell commands issued by users. For the most part, a simple mapping from a UID to a user ID, or a GID to a group name, is requested for display in the output of various commands such as ls. If the user is not authorized to retrieve the data, a UID or GID is displayed instead of the user ID or group name. As such, it is not considered a violation if the POSIXOPT.QUERYDB profile denies access to the requested information, and no violation message or audit record will be created.

   If you already have POSIXOPT QUERYDB statements defined in the CP directory, you can run the RPIDIRCT EXEC to create the RACF commands that will define this profile and do the appropriate PERMITs.

   Note: RPIDIRCT defines this profile with a universal access of READ and specifically denies access to users with a POSIXOPT QUERYDB DISALLOW statement in their directory entries.

See “Processing OpenExtensions Statements in the CP Directory” on page 49 for more information.

6. Establish auditing options for OpenExtensions. See *z/VM: RACF Security Server Auditor's Guide* for information on file-level and class-level auditing options.
7. Activate ESM support for BFS service machines.

   To use RACF for BFS permission checking and auditing, you must do the following for each BFS server:

   a. Specify ESECURITY as a start-up parameter in the DMSPARMS file. This file usually resides on the A-disk of each server and is named *server-id* DMSPARMS, where *server-id* is the user ID of the server. See *z/VM CMS File Pool Planning, Administration, and Operation* for more information.
   b. Create or modify DMSESM PROFILE based on your installation needs.

      A file called DMSESM PROFILE tells the BFS file pool server which types of authorization checking will be routed to an external security manager (ESM). This step describes what this file should look like when RACF is the designated ESM for BFS.

      If your file pool server also contains SFS files and directories that are protected by RACF, see “Setting Up DMSESM PROFILE for RACF SFS Protection” on page 187.

      See *z/VM CMS File Pool Planning, Administration, and Operation* for details about each parameter specified in DMSESM PROFILE.

      Figure 16 on page 197 shows the DMSESM PROFILE file shipped with z/VM.

      ```
      P0 DMSSECIT
      A0012 DMSUAUTH B0000 DMSAAUTH C00 DMSOAUTH D1 DMSSECIT
      A002301 A002401
      ```

      Figure 16. DMSESM PROFILE: Supplied with z/VM

      If you want RACF to protect BFS you must modify this file. To specify that RACF will be called for protection of BFS, add E1 DMSPERM to the end of line 2 in the DMSESM PROFILE supplied with z/VM. Figure 17 on page 197 shows the updated file:

      ```
      P0 DMSSECIT
      A0012 DMSUAUTH B0000 DMSAAUTH C00 DMSOAUTH D1 DMSSECIT E1 DMSPERM
      A002301 A002401
      ```

      Figure 17. DMSESM PROFILE: Modified for RACF Protection of BFS

      Place the file on a disk that is accessible to each BFS server.
   c. Replace the z/VM default BFS security exit with the exit provided by RACF.

      i) Copy ESMLIB CSLLIB from the RACF 305 disk to the BFS server's A-disk.

      ii) Add the following command to the BFS server's PROFILE EXEC before the FILESERV START command:

      ```
      RTNLOAD DMSPERM (FROM ESMLIB
      ```

      See *CMS File Pool Planning, Administration, and Operation* for details on the external security manager exit.
   d. Update the RACF SERVMACH file to identify the user ID of the RACF service machine that you want to be used for this BFS server.

      All OpenExtensions audit records created by the BFS server are logged in the SMF DATA file of the specified RACF service machine. You can optionally dedicate a RACF service machine to a BFS server or group of BFS servers. See *z/VM: RACF Security Server System Programmer's Guide* for information about dedicating a RACF service machine.

      The RACF SERVMACH file is placed on the CMS Y-disk during RACF installation, and the default is to send OpenExtensions audit records to RACFVM. If you want OpenExtensions audit records logged on a RACF service machine other than RACFVM, copy the RACF SERVMACH file to another minidisk accessed by the BFS server and change the user ID to that of the RACF service machine desired.

8. Activate ESM support for the z/VM operating system.

   To do this, specify your installation's value for NGROUPS_MAX in the ICHNGMAX macro in HCPRWA.

   The POSIX constant NGROUPS_MAX defines how many GIDs will be associated with a process for the purpose of authority checking. The value you specify becomes the NGROUPS_MAX value for the duration of the IPL.

   For instructions on coding the ICHNGMAX macro, see *z/VM: RACF Security Server Macros and Interfaces*.

   After you have updated ICHNGMAX and re-IPLed, VM will call RACF to:

   - Retrieve OpenExtensions user and group information from RACF (rather than the CP directory):
     - When a user enters the system
     - In response to queries issued by CMS
   - Authorize and audit changes to the identity of an OpenExtensions process
   - Authorize and audit attempts to execute set-UID and set-GID files in the BFS.

Defining OpenExtensions Users

On an OpenExtensions system, users are defined to the system by a *user identifier* (UID). A *UID* corresponds to a z/VM user ID. POSIX allows for the definition of character representations of UIDs, called *user names,* to be associated with a UID. On z/VM, the user name associated with a UID is defined as its z/VM user ID in lower case. This is simply the convention z/VM uses in returning a user name to a program, or displaying the user name on the screen. There are no additional steps that need to be taken to define user names to either z/VM or RACF.

The same UID value can be assigned to multiple users (allowed by the POSIX standard), either explicitly or by default, but this is not recommended. If it is done, it should be done with care because the UID is used in OpenExtensions security checks and control at an individual user level would be lost. That is, users with the same UID value would be treated as a single user during POSIX security checks. In addition, when querying RACF for information associated with a particular UID, it is unpredictable which user name the returned information will be associated with.

The OpenExtensions information for a user is defined in the OVM segment of a USER profile. You can use the ADDUSER, ALTUSER, and LISTUSER commands to define and display OpenExtensions z/VM attributes for users in the OVM segment. Only users with SPECIAL authority or authority through field-level access checking are allowed to issue the commands specifying the OVM keyword.

For more information about field-level access checking, see "Setting Up Field-Level Access for the OVM Segment" on page 203.

The OpenExtensions attributes that can be defined in the OVM segment for a user are:

- UID: an integer value that identifies a user. If no UID value exists, RACF returns a value of 4 294 967 295 (X'FFFFFFFF') to z/VM. VM treats this value as the default UID. LISTUSER will return a value of NONE for this case.
- Home directory: the user's initial directory in the OpenExtensions file system. This becomes the current working directory for the user's process when he or she first uses an OpenExtensions function. If no home directory value exists, RACF defers to the value in the CP directory.
- Initial user program: the path name of the default program to be given control when a CMS user uses the OPENVM SHELL command to become an OpenExtensions user. It is usually a shell program. This program is not used when other non-OpenExtensions programs invoke an OpenExtensions service and become OpenExtensions processes. If no initial program value exists, RACF defers to the value in the CP directory.
- File system root: identifies the byte file system that contains the user's directory hierarchy. If no file system root value exists, RACF defers to the value in the CP directory.

RACF allows path name parameters to be up to 1023 mixed-case characters.

Assigning UIDs

The easiest way to assign UIDs to your RACF users is to run z/VM's DIRPOSIX utility against the CP directory, and then run RACF's RPIDIRCT EXEC. This will result in unique UIDs being assigned to every user ID.

For details on DIRPOSIX and RPIDIRCT, see "Processing OpenExtensions Statements in the CP Directory" on page 49.

Using the ADDUSER Command

You can use the ADDUSER command to add a new OpenExtensions user. See *z/VM: RACF Security Server Command Language Reference* for more information about the ADDUSER command.

ADDUSER Example

Suppose you want to add a new OpenExtensions user. The user profile will be owned by the system administrator's user ID, SYSADM, and will be a member of the existing group SYSOM which is associated with a GID. Enter:

```
ADDUSER PJWELLS DFLTGRP(SYSOM) OWNER(SYSADM) NAME('PATRICK J. WELLS')
 OVM(UID(27) HOME(/u/pjwells) PROGRAM(/bin/sh))
```

Using the ALTUSER Command

You can use the ALTUSER command to change information about an OpenExtensions user. See *z/VM: RACF Security Server Command Language Reference* for more information about the ALTUSER command.

ALTUSER Example

To make existing OpenExtensions user CJWELLS a superuser and delete the PROGRAM from CJWELLS's profile so that the default shell program will be used when CJWELLS enters the OPENVM SHELL command, enter:

```
ALTUSER CJWELLS OVM(UID(0) NOPROGRAM)
```

Note: See *z/VM: RACF Security Server Command Language Reference* for information you should consider before you change a UID.

Using the LISTUSER Command

Use the LISTUSER command to list the details of the user profile. The details RACF lists from the OVM segment are the OpenExtensions user's:

- User identifier
- Initial directory path name
- Program path name
- File system root name.

LISTUSER Examples

To list OVM segment information for CJWELLS, enter:

```
LISTUSER CJWELLS OVM NORACF
```

The LISTUSER command displays:

```
 USER=CJWELLS

 OVM INFORMATION
 ---------------
```

```
  UID= 0000000024
  HOME= /u/cjwells
  PROGRAM= /u/cjwells/bin/myshell
  FSROOT= /../VMBFS:SERVER8:CJWELLS/
```

If the OVM segment does not exist, the LISTUSER command displays:

```
USER=CJWELLS

NO OVM INFORMATION
```

User Identity for an OpenExtensions Process

When a user ID logs on to z/VM, RACF performs user identification and authentication and, if successful, sends the OpenExtensions information contained within the user's USER profile back to CP, where the user's OpenExtensions security environment is maintained by z/VM. The UID from the OVM segment in the USER profile for the current user is used as the real, effective, and saved set-UID for the first OpenExtensions process created by the user.

OpenExtensions processes may have their UIDs changed during execution through the use of such system calls as `setuid()` and `seteuid()`. Any process subsequently created will inherit the UID values from its parent process. The effective UID of a process is used to make access check decisions for OpenExtensions resources. The z/VM user ID for the current user continues to be used to make access decisions for other z/VM resources.

Defining Superusers

POSIX 1003.1 defines a number of services that are allowed to pass security checks or perform special functions if the caller has appropriate privileges. In OpenExtensions, *appropriate privilege* is defined as superuser authority. A *superuser* is defined as a user with a UID equal to zero. A superuser is authorized to access all OpenExtensions protected objects and to issue virtually all security-relevant POSIX functions. The UID is not used when making existing (non-OpenExtensions) access decisions for z/VM resources (such as minidisks and SFS files) so the superuser privilege is confined to the OpenExtensions environment.

You should be very careful about who is given a UID of 0 as it allows almost unlimited access to OpenExtensions files and privileged functions.

Although assigning the same UID to multiple users is not generally recommended, it may be necessary in the case of superusers. You can quickly determine who your superusers are by looking at the access list of the mapping profile for UID 0:

```
RLIST VMPOSIX U0 ALL
```

For a detailed description of the mapping profiles and what they are used for, see “VMPOSIX Mapping Profiles for UIDs and GIDs” on page 206.

These mapping profiles should not be altered. Permitting a user to this profile will **not** result in a UID being assigned to the OVM segment of the USER profile.

The only OpenExtensions service that is not available to a superuser is the chaudit() (change audit) service when used to change the auditor audit options of a BFS file or directory. To change the auditor audit options, the user must have the RACF AUDITOR attribute. Superuser authority is not sufficient. A user can have both superuser and AUDITOR authority. Access to functions and resources by superusers are subject to auditing.

For details on auditing, see *z/VM: RACF Security Server Auditor's Guide*.

Defining a Single Superuser

If your OpenExtensions environment is such that you never need more than one superuser logged on at a given time, consider defining a shared user ID for use as your superuser. For example:

```
ADDUSER SUPERUSR OVM(UID(0))
RDEFINE SURROGAT LOGONBY.SUPERUSR UACC(NONE)
PERMIT LOGONBY.SUPERUSR ID(userid) CLASS(SURROGAT) ACCESS(READ)
SETROPTS CLASSACT(SURROGAT)
```

where *userid* is a user who will LOGON to the superuser user ID using the BY option of the LOGON command.

See "Defining Shared User IDs" on page 79 for more information about shared user IDs.

Defining OpenExtensions Groups

On an OpenExtensions system, groups are defined to the system by a *group identifier* (GID). A *GID* corresponds to a RACF group. POSIX allows for the definition of character representations of GIDs, called *group names,* to be associated with a GID. In RACF, the group name is simply the name of the RACF GROUP profile which contains the GID. Although the CP directory in z/VM allows the definition of mixed-case group names, RACF GROUP profiles may only exist in upper case.

The same GID value can be assigned to multiple groups (allowed by the POSIX standard), either explicitly or by default, but this is not recommended. If it is done, it should be done with care because the GID is used in OpenExtensions security checks and control at an individual group level would be lost. That is, groups with the same GID value would be treated as a single group during OpenExtensions security checks. In addition, when querying RACF for information associated with a particular GID, it is unpredictable which group name the returned information will be associated with. The GID for a group is defined in the OVM segment of a GROUP profile. You can use the ADDGROUP, ALTGROUP, and LISTGRP commands to define and display the GID for a group. Only users with SPECIAL authority or authority through field-level access checking are allowed to issue the commands specifying the OVM keyword.

The OpenExtensions attributes that can be defined in the OVM segment for a group are:

- GID: an integer value that identifies a group. If no GID value exists, RACF returns a value of 4 294 967 295 (X'FFFFFFFF') to z/VM. VM treats this value as the default GID.

Assigning GIDs

If you already have OpenExtensions groups defined in the CP directory, you can run the RPIDIRCT EXEC against it to create the equivalent RACF commands. You should consider this carefully before doing so. Unless the directory groups were defined with your existing RACF group structure in mind, then simply moving these group definitions into the RACF database may not have the results you desire. If it is not appropriate in your case to use RPIDIRCT, then you will need to manually assign GIDs to your RACF GROUPS, or write a program for this purpose. The output from the database unload utility may be helpful in identifying the RACF groups which are defined on the database.

For details on the database unload utility, see Chapter 18, "The RACF Database Unload Utility (IRRDBU00)," on page 269.

For details on RPIDIRCT, see "Processing OpenExtensions Statements in the CP Directory" on page 49.

Using the ADDGROUP Command

You can use the ADDGROUP command to add a new OpenExtensions group. See *z/VM: RACF Security Server Command Language Reference* for more information about the ADDGROUP command.

ADDGROUP Example

Suppose you want to add a new OpenExtensions group. The group profile will be owned by the system administrator's user ID, SYSADM. Enter:

```
ADDGROUP OVMIS4ME OWNER(SYSADM) OVM(GID(4))
```

Using the ALTGROUP Command

You can use the ALTGROUP command to change information about an OpenExtensions group. See *z/VM: RACF Security Server Command Language Reference* for more information about the ALTGROUP command.

ALTGROUP Example

To give group VMIS4ME a new GID, enter:

```
ALTGROUP VMIS4ME OVM(GID(3243))
```

Using the LISTGRP Command

The LISTGRP command lists the details of the group profile. For the OVM segment, the LISTGRP command displays the OpenExtensions group's group identifier (GID).

LISTGRP Example

To request the listing of the OVM segment for group OVMIS4ME, enter:

```
LISTGRP OVMIS4ME OVM NORACF
```

The LISTGRP command displays:

```
    INFORMATION FOR GROUP OVMIS4ME
      OVM INFORMATION
         GID= 0000003243
```

Group Identity for an OpenExtensions Process

When a user logs on to z/VM, RACF sends GID information contained within GROUP profiles back to the z/VM control program (CP). RACF returns the GID associated with the user's *primary group,* which is the user's default group as specified in the USER profile.

In addition to group names, OpenExtensions also allows for the definition of a *supplementary group* (SGID) list, which is a list of GIDs associated with the groups the user is connected to. The supplementary group list corresponds to list-of-groups processing in RACF. If SETROPTS GRPLIST is in effect, RACF returns the supplementary group list to CP, up to the value defined for NGROUPS_MAX on the ICHNGMAX macro in HCPRWA.

The GID from the OVM segment in the user's primary group is used as the real, effective, and saved set-GID for the first OpenExtensions process created by the user. OpenExtensions processes may have their GIDs changed during execution through the use of such system calls as `setgid()` and `setegid()`. Any process subsequently created will inherit the GID values from its *parent process*. The effective GID and supplementary group list are used to make access checking decisions for OpenExtensions resources.

Supplementary Group Lists

The supplementary group list is created only if SETROPTS GRPLIST is in effect, which means that list-of-groups processing is active in RACF. The supplementary group list cannot contain duplicate entries. If multiple groups share the same GID, which could be the default—4 294 967 295 (X'FFFFFFFF')—they are represented by a single entry. Thus, more groups than the value specified by NGROUPS_MAX can be represented in the supplementary group list.

These groups generally do not change until the user logs off and back on, though use of the `newgrp()` utility may result in a group being added to the SGID list.

Setting Up Field-Level Access for the OVM Segment

To let a user see or change OVM fields in a RACF user profile, you can set up field-level access. You can authorize a user to specified fields in the user's own profile. Generally speaking, you will want to prevent users from changing their UIDs, but will want to allow them to change their HOME, PROGRAM, and FSROOT fields. To authorize users to their OVM fields in this manner:

1. Define a profile for each of the OVM fields with a RACF RDEFINE command. Figure 18 on page 203 shows commands for the four OpenExtensions fields.

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

Figure 18. RDEFINE Commands for RACF OVM User Profile Fields

2. Permit users to access the fields with RACF PERMIT commands. Figure 19 on page 203 shows commands for the four fields.
   - &RACUID lets all users look at their own fields.
   - READ access lets users read the UID field.
   - UPDATE access lets users change:
     - Their home directory in the HOME field
     - The program invoked for a z/VM OPENVM SHELL command in the PROGRAM field
     - Their file system root directory in the FSROOT field.

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

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

Figure 19. PERMIT Commands for RACF OVM User Profile Fields

3. Activate the FIELD class with the RACF SETROPTS command; see Figure 20 on page 203.

```
SETROPTS CLASSACT(FIELD)
```

Figure 20. SETROPTS Command

See *z/VM: RACF Security Server Command Language Reference* for the other parameters on the RDEFINE, PERMIT, and SETROPTS commands.

OpenExtensions BFS File Security

In z/VM, the POSIX file system is called the *byte file system (BFS).* The BFS is implemented in OpenExtensions z/VM as an extension of the *shared file system (SFS).*

The files within the file systems are managed by the BFS server. Security information is carried along with the file rather than in a RACF profile. Security information consists of such items as the file owner UID and GID, the file's permission bits, and audit settings. BFS calls an ESM exit to perform access decisions. The access rules RACF that enforces are POSIX 1003.1 access controls, which use permission bits.

There are three different sets of users who might want to access a file:

- The owner of the file
- The members of the owning group
- All other users on the system

These users might want four different kinds of access permission (of which only three apply to any given file):

- Read
- Write
- Search (directories only)
- Execute (files other than directories)

OPENVM protects its data by means of file *permission bits.* Every file has a three-part string of file permission bits associated with it. The three parts represent the owner, group, and others, and one bit within each part represents each type of access. The owner of the file sets the bits on or off to grant or deny permission to access the file. RACF recognizes these bits and grants or denies access accordingly.

The ESM exit is also called to authorize various functions that alter security information for a file. For example, if a user wishes to change the owner of a file, or the permission bits associated with a file, the ESM exit running in BFS will be called to authorize the function (and perform auditing if necessary).

The permission bits can be changed using the `chmod` command. The file owner UID can be changed using the `chown()` command and the file owner GID can be changed using the `chgrp()` command. For a general discussion on file security, see *OpenExtensions for z/VM User's Guide*. For information about OpenExtensions commands, see *OpenExtensions for z/VM Command Reference*.

The authorization checks that the BFS server makes in the absence of an external security manager (ESM) are documented in *z/VM CMS File Pool Planning, Administration, and Operation*. For the most part, RACF enforces the same rules, because the POSIX standard is being enforced. There are some exceptions.

- BFS allows the owner and auditor audit options contained within a BFS file or directory to be modified, but does not produce any audit records as a result of the audit options. RACF will produce audit records based on the file level audit options, depending on the class wide settings for the RACF OpenExtensions auditing classes (see *z/VM: RACF Security Server Auditor's Guide*).
- RACF allows a user with the RACF AUDITOR attribute to enforce the installation's auditing policy in the OpenExtensions environment. A RACF auditor can modify the auditor audit options in any BFS file or directory. Further, a RACF auditor has automatic READ and SEARCH access to any BFS directory; authority through the permission bits associated with the directory is not necessary.
- RACF allows a further level of protection for set-UID and set-GID files beyond requiring execute access to the file through the file permission bits. See “Protecting Set-UID and Set-GID Executable Files” on page 205 for details.
- BFS considers a user with SFS administrator authority to have appropriate privileges. That is, an SFS administrator is considered to be a superuser. RACF does not consider an SFS administrator to be a superuser.

Changing the Identity of an OpenExtensions Process

In POSIX, there are defined methods of changing the UIDs and GIDs (real, effective, and saved set) of a running process, which means it will run with different access authorities. The functions setuid() and seteuid() can change UIDs associated with a process. Similarly, setgid() and setegid() can change the GIDs associated with a process, as can the newgrp() utility. This could be useful for example to make it possible for an authorized ***daemon*** process to create processes which are associated with other login UIDs.

The POSIX standard defines what authority is required to perform these privileged functions, and the results the function should have upon successful completion. For example, setuid() changes the effective UID of a process to the one specified only if the requesting UID is a superuser, or if the specified UID

is the same as the real or saved set-UID of the process. Further, if the requester is a superuser, then all three of the process' UIDs (real, effective and saved set) are changed to the value specified. This would affect the way subsequent setuid() and seteuid() calls work.

RACF enforces the rules of the POSIX standard for these functions, just as z/VM would in the absence of an ESM. For newgrp, which is an extension of the POSIX standard, RACF will allow a user to switch to the new group if the user is connected to the GROUP profile in the RACF database. RACF will also audit these functions depending on the SETROPTS LOGOPTIONS setting for the PROCESS class. See *z/VM: RACF Security Server Auditor's Guide* for details.

z/VM will not call RACF to authorize or audit changes to the identity of an OpenExtensions process unless the ICHNGMAX macro in HCPRWA has been coded with a valid NGROUPS_MAX value. See "Setting up Support for OpenExtensions" on page 195 for details.

Protecting Set-UID and Set-GID Executable Files

A *set-UID* file is a BFS executable file that changes the effective UID of the process running the program to the owning UID of the file. A *set-GID* file is a BFS executable file that changes the effective GID of the process running the program to the owning GID of the file.

A superuser or the file owner can use a `chmod` shell command or `chmod()` function to change an executable file into a set-UID and/or set-GID file by turning on bits in the file mode of the file.

If one or both of these bits are *on*, the effective UID, effective GID, or both, plus the saved UID, saved GID, or both, for the process running the program are changed to the owning UID, GID, or both, for the file. This change temporarily gives the process running the program access to data the file owner or group can access.

In a new file, both bits are set *off*. Also, if the owning UID or GID of a file is changed or if the file is written in, the bits are turned *off*.

As with any other executable file within BFS, a set-UID or set-GID program may not be run for a user unless that user has execute access to the file containing the program. For details on file permissions, see "OpenExtensions BFS File Security" on page 203.

When a user executes a set-UID or set-GID file, the file is copied to the user's address space for execution. As such, the user has the ability to trace and modify the program as it is executing. The user would also then be able to pass control off to another program which would then continue to execute under the authority of the original file owner. Such a program would be able to access BFS files based on the file owner's access privileges. In this regard, set-UID and set-GID files create security-sensitive processes on z/VM.

To further restrict the ability to execute set-UID and set-GID files beyond requiring execute access to the file, z/VM provides the POSIXOPT EXEC_SETIDS ALLOW directory control statement to specify which users are allowed to have their effective UID/GID changed during exec(). This statement would be placed into the directory entry of any user you want to be able to execute **all** set-UID and set-GID files for which they have execute permission (note that the system configuration file also contains a statement which indicates the system-wide default value to be used if a user's directory entry does not contain a POSIXOPT EXEC_SETIDS statement).

RACF will allow a further level of granularity to protect exec() processing beyond the all-or-none ability of POSIXOPT EXEC_SETIDS ALLOW. Define profiles in the VMPOSIX class of the format EXEC.U*uid* and EXEC.G*gid*, where *uid* is the file owner's UID and *gid* is the file owner's GID. A user with READ access to this profile (or these profiles) is authorized to execute a program running under the UID and/or GID of the owner of the program (depending on whether the set-UID bit, set-GID bit, or both, is set in the file mode of the file being executed). For example, to allow user ID SUSAN to execute a file which will run with a UID of 30:

```
RDEFINE VMPOSIX EXEC.U30 UACC(NONE)
PERMIT EXEC.U30 CLASS(VMPOSIX) ACCESS(READ) ID(SUSAN)
```

The VMPOSIX profiles are checked **in addition** to the check for execute access to the file. It is not necessary to permit superusers to these profiles. If the profile does not exist, or the VMPOSIX class is not active, RACF will fail the exec().

In order for z/VM to give control to RACF to authorize the exec of a set-UID or set-GID file in BFS, control must be on in your VMXEVENT profile for the DIAG280 event, which is the mechanism by which exec is implemented on z/VM. By default, DIAG280 will be controlled when you install RACF on z/VM. If you do not wish to enforce exec protection in the VMPOSIX class, then turn off control for DIAG280 in your VMXEVENT profile.

```
RALTER VMXEVENT profile-name ADDMEM(DIAG280/NOCTL)
SETEVENT REFRESH profile-name
```

This will avoid the performance overhead associated with the call to RACF. If control is off for DIAG280, z/VM will check the POSIXOPT settings in the CP directory to determine if the user is authorized to execute the set-UID or set-GID file. For details on how to control events in your VMXEVENT profile, see “Creating VMXEVENT Profiles” on page 129..

These VMPOSIX profiles provide the ability to restrict execute access to set-UID and set-GID files in BFS at an individual user or group level. In contrast, using only the ***other*** execute access permission bits to restrict usage of the set-UID or set-GID file corresponds to using UACC(READ) in a RACF profile. Also, the ability to execute set-UID files may be separated from the ability to execute set-GID files. For details on file permissions, see “OpenExtensions BFS File Security” on page 203.

VMPOSIX Mapping Profiles for UIDs and GIDs

For each UID that has been defined in the OVM segment of a USER profile, a VMPOSIX profile called U*uid* exists. The access list of a U*uid* profile contains all user IDs that have been assigned this UID.

For each GID that has been defined in the OVM segment of a GROUP profile, a VMPOSIX profile called G*gid* exists. The access list of a 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/VM or application programs.

RACF maintains these mapping profiles automatically when UIDs and GIDs are added, changed, or deleted.

For example, if the following command is issued

```
ADDUSER BRUCE OVM(UID(13))
```

RACF creates a VMPOSIX profile named U13 with BRUCE contained on the access list. If the following command is subsequently issued

```
ALTUSER BRUCE OVM(UID(55))
```

RACF deletes the U13 profile and creates a U55 profile with BRUCE contained on the access list.

In general, *you should not alter these profiles.* However, it is possible 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, VM 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/VM a return code indicating that the UID or GID is invalid.

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 OpenExtensions z/VM. For example, if the user name is BRUCE, enter:

```
LISTUSER BRUCE OVM NORACF
```

to display the UID associated with BRUCE. If, for example, LISTUSER displays a UID of 13, you would then enter:

```
RDEFINE VMPOSIX U13 UACC(NONE)
PERMIT U13 CLASS(VMPOSIX) ACCESS(NONE) ID(BRUCE)
PERMIT U13 CLASS(VMPOSIX) ID(your-userid) DELETE
```

The second PERMIT command is necessary because RDEFINE puts the profile creator on the access list.

If you are unable to determine the user name or group name from z/VM, look at the output from the database unload utility 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.

Chapter 13. Controlling Access to Privileged RACF Functions

Additional CP controls in RACF module HCPRWA include these macros for customization:

- RACSERV - used to define the name of the RACF service machine
- GLBLDSK - used to create a list of minidisks in a global minidisk table
- SYSSEC - used to establish the relationship between RACF's response to access requests and the final disposition of the requests by the Control Program (CP)

See *z/VM: RACF Security Server Macros and Interfaces* for details.

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 to have access to particular profiles to perform specific tasks. For example, READ or UPDATE access to profile ICHCONN allows a user to issue certain RACROUTE requests on z/VM. There are many other profiles used by many products.

You will probably activate the FACILITY class for the first such profile that is required on your system. You will probably create FACILITY profiles as needed to control who can use a number of processes on your system.

It is also recommended 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 114.

```
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 the FACILITY profiles used by the function.
- Create a group representing the function and make the user group-SPECIAL 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.

  For more information about the GENERICOWNER option, see "Restricting the Creation of General Resource Profiles (GENERICOWNER Option)" on page 103.

Controlling Which Virtual Machines Can Issue RACROUTE Requests

You can use RACF to control which virtual machines can issue RACROUTE requests. This protection applies to all RACROUTE requests that specify RELEASE=1.9 or any later release. (This protection does

not apply to RACROUTE requests issued within the RACF service machine.) To do this, see the procedure in *z/VM: Security Server RACROUTE Macro Reference*.

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 UID field of the OVM segment of user profiles. If you give a user UPDATE authority to this profile, the user can modify the UID 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, that is, the OVM segment of user and group 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:

   - USER for user profiles
   - GROUP for group profiles
   - Class name for general resource profiles.

   segmentname
   is one of the following:

   - OVM for OVM segments
   - BASE for BASE segments (this is supported only by user-written code).

   Note:

   a. This operand is also used on RACF commands to work with the segment.
   b. See "Setting Up Field-Level Access for the OVM Segment" on page 203 for examples of field-level access checking for OVM segments.

   fieldname
   is the name of the field to be protected as described in Table 30 on page 211.

   When you specify a UACC of NONE, you prevent all users from accessing the OVM 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 OVM segment for all user profiles.

Attention:
Note that the profile name USER.OVM.* 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 defining the generic profile.

To control access to specific fields in the OVM segment of user profiles, issue the RDEFINE command and specify the specific field as the third qualifier in the profile name. Use Table 30 on page 211 to determine which qualifier to use. For example, when changing the account number field in a OVM segment, users specify the UID suboperand on the OVM operand of the ALTUSER command:

```
ALTUSER userid OVM(UID(account-number))
```

According to Table 30 on page 211, to control access to the UID suboperand, create a profile using the OVMADM qualifier:

```
RDEFINE  FIELD  USER.OVM.OVMADM  UACC(NONE)
```

2. Allow specific users or groups to have the appropriate access to the FIELD profile:

   ```
   PERMIT USER.OVM.UID CLASS(FIELD) ID(OVMADM)
          ACCESS(UPDATE)
   ```

   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 UID field in the OVM segment of their own user profiles.

   ```
   PERMIT USER.OVM.UID 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, only SPECIAL users can access fields in segments (other than the base segment) of RACF profiles.

4. It is recommended that you activate SETROPTS RACLIST processing for the FIELD 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 114.

   ```
   SETROPTS  RACLIST(FIELD)
   ```

   Note: If 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 30. Relationship of RACF Command Suboperands to FIELD Profile Names

OVM Segment in:	**To control the use of this suboperand:**	**Use this qualifier in FIELD profiles:**
User Profiles	UID HOME PROGRAM FSROOT	UID HOME PROGRAM FSROOT
Group Profiles	GID	GID

Command Summary and Authority Required to Issue RACF Commands

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 31 on page 212 shows, in alphabetic order, each of the commands and its functions, and the system(s) on which you can invoke it. For a complete description of the authority required to issue a command and its operands, see *z/VM: RACF Security Server Command Language Reference*.

Table 31. Functions of RACF Commands

RACF Command	Command Functions	System
ADDDIR	• RACF-protect by a discrete or generic profile one or more SFS directories.	z/VM
ADDFILE	• RACF-protect by a discrete or generic profile one or more SFS files.	z/VM
ADDGROUP	• Define one or more new groups as a subgroup of an existing group. • On z/OS, specify a model data set profile for a group. • On z/OS, define default DFP information for a group. • On z/VM, define the OpenExtensions z/VM information for a group.	z/OS, z/VM
ADDSD [1]	• 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 DATASET profiles. • Create a new data set model profile.	z/OS
ADDUSER	• Define one or more new users and connect the users to their default connect group. • Define a password, password phrase, both, or neither for one or more users. • On z/OS, specify a model data set profile for a user. • On z/OS, define CICS® operator information. • On z/OS, define default DFP information for a user. • On z/OS, define the preferred national language. • On z/OS, define default operator information. • On z/VM, define the OpenExtensions z/VM information for a user. • On z/OS, define default TSO logon information for a user. • On z/OS, define default work attributes.	z/OS, z/VM
ALTDIR	• Change a discrete or generic SFS directory profile.	z/VM
ALTDSD [1]	• Change one or more discrete or generic DATASET 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.	z/OS
ALTFILE	• Change a discrete or generic SFS file profile.	z/VM
ALTGROUP	• Change the information in one or more group profiles (such as the superior group, owner, or model profile name). • On z/OS, change or delete the default DFP information for a group. • On z/VM, add, change, or delete the information for an OpenExtensions z/VM group.	z/OS, z/VM

Table 31. Functions of RACF Commands (continued)

RACF Command	Command Functions	System
ALTUSER	• Change the information in one or more user profiles (such as the owner, universal access authority, or security level). • Change the password or password phrase of one or more users. • 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. • On z/OS, change or delete CICS operator information. • On z/OS, change or delete the default DFP information for a user. • On z/OS, change the preferred national language. • On z/OS, change or delete the default operator information. • On z/VM, add, change, or delete the information for an OpenExtensions z/VM user. • On z/OS, change or delete the default TSO logon information for a user. • On z/OS, change or delete the default work attributes.	z/OS, z/VM
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.	z/OS, z/VM
DELDIR	• Remove RACF-protection from one or more SFS directories.	z/VM
DELFILE	• Remove RACF-protection from one or more SFS files.	z/VM
DELDSD [1]	• Delete one or more discrete or generic DATASET profiles. • Delete a discrete DATASET 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.	z/OS
DELGROUP	• Delete one or more groups and their relationship to the superior group.	z/OS, z/VM
DELUSER	• Delete one or more users and remove all their connections to RACF groups.	z/OS, z/VM
END	• Terminate a RACF command session.	z/VM
HELP	• Display the function and proper syntax of RACF commands. • Display an explanation of command-related messages.	z/OS, z/VM
LDIRECT	• List the details of one or more discrete or generic DIRECTRY profiles, including the users and groups authorized to access an SFS directory.	z/VM
LFILE	• List the details of one or more discrete or generic FILE profiles, including the users and groups authorized to access the SFS file.	z/VM
LISTDSD [1]	• List the details of one or more discrete or generic DATASET profiles, including the users and groups authorized to access the data sets. • Determine the most specific matching generic profile for a data set.	z/OS
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 OVM) of the group profile.	z/OS, z/VM
LISTUSER	• List the details of one or more user profiles, including all the groups to which each user is connected. • List only the information contained in a specific segment (for example, DFP or OVM information) of a user profile.	z/OS, z/VM

Table 31. Functions of RACF Commands (continued)

RACF Command	Command Functions	System
PASSWORD or PHRASE	• Change your own user password or password phrase. • Change one or more users' change interval for passwords and password phrases. • Remove one or more user passwords.	z/OS, z/VM
PERMDIR	• Give or remove authority to access an SFS directory to specific users or groups. • Change the level of access authority to a directory for specific users or groups. • Copy the list of authorized users from one directory profile to another. • Delete an existing access list.	z/VM
PERMFILE	• Give or remove authority to access an SFS file to specific users or groups. • Change the level of access authority to a file for specific users or groups. • Copy the list of authorized users from one file profile to another. • Delete an existing access list.	z/VM
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.	z/OS, z/VM
RACF	• Begin a RACF command session.	z/VM
RALTER	• Change the discrete and/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.	z/OS, z/VM
RDEFINE	• RACF-protect by a discrete and/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.	z/OS, z/VM
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.	z/OS, z/VM
REMOVE	• Remove one or more users from a group and assign a new owner for any group data sets owned by the users.	z/OS, z/VM
RLIST	• List the details of discrete and/or generic profiles for one or more resources whose class is defined in the class descriptor table.	z/OS, z/VM
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.	z/OS, z/VM
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.	z/OS, z/VM

Table 31. Functions of RACF Commands (continued)

RACF Command	Command Functions	System
SETEVENT	• Change the auditing or controlling of z/VM events. • Prevent users from issuing the DIAL, UNDIAL, and MESSAGE commands before logging on to the system. • Display the current status for z/VM events.	z/VM
SETRACF **2**	• Deactivate and reactivate RACF.	z/VM
SETROPTS	Dynamically set system-wide options relating to resource protection, specifically: For both z/OS and z/VM systems: • Choose the resource classes that RACF is to protect. • Gather and display RACF statistics. • Set the universal access authority (UACC) for terminals. • 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 checking for previous passwords and password phrases. • Limit unsuccessful attempts to access the system using incorrect passwords and password phrases. • Control maximum change interval for passwords and password phrases. • Control mixed-case passwords. • Enable the use of special characters in passwords. • Enable the KDFAES password algorithm. • Warn of password and password phrase expiration. • Control global access checking for selected individual resources and/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 whether a profile creator's user ID is automatically added to the profile's access list. For z/OS systems only: • 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.	z/OS, z/VM z/OS

Table 31. Functions of RACF Commands (continued)

RACF Command	Command Functions	System
SMF [3]	• Restart SMF recording. • Switch to the alternate SMF file.	z/VM
SRDIR	• List the SFS directory profile names that meet search criteria.	z/VM
SRFILE	• List the SFS file profile names that meet search criteria.	z/VM

Notes:

1. You can issue the SETRACF command only from a RACF service machine. A RACF service machine can run disconnected, thereby allowing a secondary console to issue this command.

   By default, RACF sets up the OPERATOR as the secondary console for a RACF service machine; the OPERATOR can issue the command to deactivate RACF. For example,

   ```
   SEND RACFVM SETRACF INACTIVE
   ```

   If you issue SETRACF for any RACF service machine in a multiple service machine environment, it will apply to all service machines.
2. SMF is only issued with SMSG.

Summary of Authorities and Commands

This section summarizes the attributes and authorities that can be assigned to users, and the z/VM RACF commands and operands that can be issued for each authority. There are eight tables which give information in four major categories: user attributes, group authorities, access authorities, and miscellaneous authorities.

Table 32. z/VM Commands and Operands You Can Issue If You Have the SPECIAL or group-SPECIAL Attribute

Command	Operands
ADDDIR	with all operands
ADDFILE	with all operands
ADDGROUP	with all operands
ADDUSER	with all operands, but for group-SPECIAL user only when user also has CLAUTH(USER)
ALTDIR	with all operands except GLOBALAUDIT
ALTFILE	with all operands except GLOBALAUDIT
ALTGROUP	with all operands
ALTUSER	with all operands except UAUDIT or NOUAUDIT NOEXPIRED can only be issued by a user with the SPECIAL attribute at the system level.
CONNECT	with all operands
DELDIR	with all operands
DELFILE	with all operands
DELGROUP	with all operands
DELUSER	with all operands
LDIRECT	with all operands
LFILE	with all operands
LISTGRP	with all operands
LISTUSER	with all operands

Table 32. z/VM Commands and Operands You Can Issue If You Have the SPECIAL or group-SPECIAL Attribute (continued)

Command	Operands
PASSWORD	with all operands
PERMDIR	with all operands
PERMFILE	with all operands
PERMIT	with all operands
RALTER	with all operands except GLOBALAUDIT
RDEFINE	with all operands
RDELETE	with all operands
REMOVE	with all operands
RLIST	with all operands
SEARCH	with all operands
SETEVENT	with all operands. If you have both the SPECIAL and AUDITOR attributes and specify the REFRESH operand, you can refresh both auditing and controlling of z/VM events on your system. If you have only the SPECIAL attribute and specify the REFRESH operand, you can refresh only controlling of z/VM events on your system. If you specify a profile name on the REFRESH operand, you must also be the owner of the profile, or have ALTER authority to the profile. Users with the group-SPECIAL attribute cannot issue this command.
SETROPTS	with all operands except AUDIT, NOAUDIT, CMDVIOL or NOCMDVIOL, APPLAUDIT, NOAPPLAUDIT, LOGOPTIONS, OPERAUDIT, NOOPERAUDIT, SAUDIT, NOSAUDIT, SECLABELAUDIT, NOSECLABELAUDIT, and SECLEVELAUDIT, NOSECLEVELAUDIT, which require the AUDITOR attribute. User with group-SPECIAL attribute can issue only REFRESH GENERIC and LIST.
SMF	with all operands. Users with group-SPECIAL attribute cannot issue this command.
SRDIR	with all operands
SRFILE	with all operands

Table 33. z/VM Commands and Operands You Can Issue If You Have the AUDITOR or group-AUDITOR Attribute

Command	Operands
ALTDIR	only with GLOBALAUDIT
ALTFILE	only with GLOBALAUDIT
ALTUSER	only with UAUDIT or NOUAUDIT
LDIRECT	with all operands, lists GLOBALAUDIT option
LFILE	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
SETEVENT	with only the LIST and REFRESH operands. If you have both the SPECIAL and AUDITOR attributes and specify the REFRESH operand, you can refresh both auditing and controlling of z/VM events on your system. If you have only the AUDITOR attribute and specify the REFRESH operand, you can refresh only auditing of z/VM events on your system. If you specify a profile name on the REFRESH operand, you must also be the owner of the profile, or have ALTER authority to the profile. Users with the group-AUDITOR attribute cannot issue this command.
SETROPTS	only with AUDIT, NOAUDIT, CMDVIOL or NOCMDVIOL, LOGOPTIONS, OPERAUDIT, NOOPERAUDIT, SAUDIT, NOSAUDIT, SECLABELAUDIT, NOSECLABELAUDIT, SECLEVELAUDIT, NOSECLEVELAUDIT, LIST, or REFRESH GENERIC
SMF	with all operands. Users with group-AUDITOR attribute cannot issue this command.
SRDIR	with all operands
SRFILE	with all operands

Table 34. z/VM Commands and Operands You Can Issue If You Have the ROAUDIT Attribute

Command	Operands
LDIRECT	with all operands, lists GLOBALAUDIT option
LFILE	with all operands, lists GLOBALAUDIT option
LISTDSD[1]	with all operands, lists GLOBALAUDIT option
LISTGRP	with all operands
LISTUSER	with all operands, lists UAUDIT or NOUAUDIT operand
RLIST	with all operands, lists GLOBALAUDIT option
SEARCH	with all operands
SETROPTS	only with LIST
SRDIR	with all operands
SRFILE	with all operands

Table 35. z/VM Commands and Operands You Can Issue If You Have the OPERATIONS or group-OPERATIONS Attribute

Command	Operands
LDIRECT	with all operands except GLOBALAUDIT
LFILE	with all operands except GLOBALAUDIT
RLIST	with all operands except GLOBALAUDIT
SEARCH	with all operands
SETROPTS	only with REFRESH
SRDIR	with all operands
SRFILE	with all operands

Table 36. z/VM Commands and Operands You Can Issue If You Have the CLAUTH or group-CLAUTH Attribute

Command	Operands
CLAUTH	
ADDUSER[1]	with all operands except OPERATIONS, NOOPERATIONS, SPECIAL, NOSPECIAL, AUDITOR, NOAUDITOR, ROAUDIT, or NOROAUDIT
ALTUSER[2]	only with CLAUTH or NOCLAUTH
RALTER[3]	only with ADDVOL
RDEFINE[4]	with all operands
SETROPTS[4]	only with REFRESH GLOBAL or REFRESH GENERIC

Notes:

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, 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.

Table 37. z/VM 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.
CONNECT	**ALTUSER** only with AUTHORITY, GROUP, 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**[1] with all operands **ADDUSER**[2] with all operands except OPERATIONS, SPECIAL, AUDITOR, or ROAUDIT **ALTGROUP**[3] only with SUPGROUP **ALTUSER** only with AUTHORITY, GROUP, or UACC **CONNECT** with all operands except SPECIAL, NOSPECIAL, OPERATIONS or NOOPERATIONS **DELGROUP**[1] with all operands **LISTGRP** only with group name **REMOVE** with all operands

Note:

1. This command applies to the superior group.
2. 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.
3. This command applies to current and new superior groups. You may have JOIN authority in one group and be owner of or be connected with the group-SPECIAL attribute to another group.

Table 38. z/VM 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

Table 38. z/VM Commands and Operands You Can Issue If You Have an Access Authority (continued)	
Access Authorities	**Commands and Operands You Can Issue If You Have This Authority**
READ UPDATE CONTROL	**LDIRECT** with all operands except ALL or AUTHUSER **LFILE** with all operands except ALL or AUTHUSER **RLIST** with all operands except ALL or AUTHUSER **SEARCH** with all operands **SRDIR** with all operands **SRFILE** with all operands
ALTER	**ALTDIR**[1] with all operands except GLOBALAUDIT or OWNER **ALTFILE1**[1] with all operands except GLOBALAUDIT or OWNER **DELDIR**[1] with all operands **DELFILE**[1] with all operands **LDIRECT**[1] with all operands **LFILE**[1] with all operands **PERMDIR**[1] with all operands **PERMFILE**[1] with all operands **PERMIT**[1] with all operands **RALTER**[1] with all operands except ADDVOL, GLOBALAUDIT, or OWNER **RDELETE**[1] with all operands **RLIST**[1] with all operands

Note:

1. This command applies to discrete profiles only.

Table 39. z/VM 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, REVOKE, RESUME, PASSWORD, NOPASSWORD, CLAUTH or NOCLAUTH **DELUSER** with all operands **LISTUSER** with all operands **PASSWORD** only with USER

Table 39. z/VM Commands and Operands You Can Issue If You Own a Profile (continued)	
Owner of RACF Profile	**Commands and Operands You Can Issue If You Are the Owner**
Owner of group profile	**ADDGROUP**[2] with all operands **ADDUSER**[3] with all operands except OPERATIONS, SPECIAL, AUDITOR, or ROAUDIT **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	**ALTDIR** with all operands except GLOBALAUDIT **ALTFILE** with all operands except GLOBALAUDIT **DELDIR** with all operands **DELFILE** with all operands **LDIRECT** with all operands **LFILE** with all operands **PERMDIR** with all operands **PERMFILE** with all operands **PERMIT** with all operands **RALTER** with all operands except GLOBALAUDIT **RDELETE** with all operands **RLIST** with all operands **SEARCH** with all operands **SRDIR** with all operands **SRFILE** with all operands

Note:

1. This command applies to CLAUTH or NOCLAUTH only if you have the CLAUTH attribute for the class to be added or deleted.
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 may 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.

Table 40. z/VM Commands and Operands You Can Issue for Miscellaneous Reasons

User ID Relationship	Commands and Operands You Can Issue for Miscellaneous Reasons
User ID is the current user	**ALTUSER** with DFLTGRP or NAME only **LISTUSER** with user ID only **PASSWORD** with INTERVAL or PASSWORD only
None	**RVARY**[1] with all operands
User ID is the second qualifier of an SFS directory name	**ADDDIR** with all operands **ALTDIR** with all operands except GLOBALAUDIT or OWNER **DELDIR** with all operands **LDIRECT** with all operands **PERMDIR** with all operands **SRDIR** with all operands
User ID is the second qualifier of an SFS file name	**ADDFILE** with all operands **ALTFILE** with all operands except GLOBALAUDIT or OWNER **DELFILE** with all operands **LFILE** with all operands **PERMFILE** with all operands **SRFILE** with all operands

Note:

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.

Chapter 14. Managing the RACF Security System

As the security administrator, you should be familiar with the operating considerations discussed in this chapter.

Coordinating Profile Updates

You should plan to update profiles so they remain consistent with other profiles on the database while making sure the updating process does not interfere with other jobs running in the system.

Each individual operation performed by RACF serializes on a RACF database, but a command or function may 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. Some of these changes that are delayed until the user logs on again are SPECIAL, OPERATIONS, AUDITOR, ROAUDIT, and the list of connected groups examined by RACROUTE REQUEST=FASTAUTH. However, a LISTUSER or LISTGRP command issued immediately after the change shows the new values.

In the following example, the system administrator inadvertently creates a situation where a profile exists, but it does not have an owner.

Example:

1. The system administrator deletes a user who is logged on.
2. That user, while logged on, defines a profile for a resource.

This creates an *ownerless* profile.

To prevent the creation of ownerless profiles, do not delete a user who is logged on. Instead, prevent the user from logging on by revoking that user with the ALTUSER command before deleting the user profile. If necessary, have the operator force the user off the system, then revoke the user, and delete the user profile when you have completed your preparations. See “Summary of Steps for Deleting Users on z/VM” on page 85.

The RAC Command Processor

Users can use the RAC command processor to issue RACF commands. The RAC command processor allows users to issue RACF commands without isolating themselves in a RACF command session.

Note: RAC cannot be issued from a RACF service machine.

For example, instead of the following sequence:

```
RACF
ADDUSER JOE
RDEFINE VMMDISK JOE.191 UACC(NONE)
END
```

a user could issue:

```
RAC ADDUSER JOE
RAC RDEFINE VMMDISK JOE.191 UACC(NONE)
```

This allows the user to issue CMS or CP commands between the RACF commands.

Also, the RAC command processor allows your installation to tailor the RACF commands so that users need not follow the syntax of the RACF commands as shipped by IBM. (For more information, see *z/VM: RACF Security Server System Programmer's Guide*.)

Using 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. These reports can help you to (1) check the initial steps you took to establish system security, and (2) make additional security checks periodically.

On z/VM systems, the reports DSMON can produce are:

- System report
- Selected user attribute reports
- Selected data sets report
- RACF exits report
- Class descriptor table report
- Global access table report
- Group tree report

A short description of each report follows. For more information on these reports and how to invoke the data security monitor, see *z/VM: RACF Security Server Security Administrator's Guide*.

System Report

This report contains information such as the identification and model of the processor complex. 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.

Selected User Attribute Reports

The selected user attribute report lists all RACF users with the SPECIAL, OPERATIONS, AUDITOR, ROAUDIT, 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 who need to be 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, ROAUDIT, 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.

Selected Data Sets Report

This report lists the names of the primary and backup RACF database(s). You can use the selected data sets report to determine if your RACF data sets are protected by RACF.

RACF Exits Report

This report lists the names of all the installation-defined RACF exit routines and specifies the size of each exit routine module.

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 could 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.

Class Descriptor Table Report

This report lists, for each general resource class, 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.

Global Access Table Report

This report lists, for each resource class in the global access table, all 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.

Group Tree Report

This report lists, for each requested group, all its subgroups, all 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).

Controlling Access to RACF Passwords and Password Phrases

Installation personnel should ensure that the security of RACF user logon passwords or password phrases is not violated. It is possible for an operator to display password or password phrase information when displaying real storage at a console. The installation should monitor the operator's activities to ensure passwords or password phrases remain secure. In addition, you should restrict access to dumps that might contain password or password phrase information.

Deactivating RACF

On z/VM, you can deactivate RACF as follows:

- You can deactivate the RACF database by using the RVARY command with the INACTIVE operand. (You must deactivate the RACF database, for example, if you want to perform maintenance on it.) In this situation, failsoft processing takes effect. Each time you request access to a resource, RACF, through failsoft processing, prompts the operator to validate your request. The operator then either grants or denies you access to the resource. (For more information on failsoft processing, see *z/VM: RACF Security Server System Programmer's Guide* and the following section of this chapter.)

  Notes:

  1. You cannot log on while the RACF database is inactive.
  2. The RVARY command is automatically propagated to all RACF servers running on the same z/VM system, and to other systems in the same SSI cluster as the issuing system. RACF does not automatically propagate the RVARY command to z/VM systems that share the RACF database outside of an SSI cluster.
- You can deactivate RACF itself by using the SETRACF command with the INACTIVE operand. (You might do this, for example, if you need to perform maintenance on RACF itself immediately after it is installed.) Deactivating RACF in this manner causes CP to handle all authorization requests as they were prior to installing RACF.

Note: When you use the SETRACF command, it applies to all service machines on that z/VM system.

- You can also deactivate RACF itself by modifying the ICHSECOP module.

Attention:
Notes: 1. Deactivating RACF by modifying ICHSECOP is not recommended for use on z/VM systems. It results in a situation where no users, including users who are already logged on, can access the system or any of its resources. 2. The RVARY INACTIVE command will cause a lockout if it is issued using RAC. Because RAC issues commands directly to the active RACF service machine, it would be unable to issue an RVARY ACTIVE command when RACF is inactive. Use of the RVARY command should be limited to a RACF command session using the RACF EXEC.

Failsoft Processing

During failsoft processing (when the RACF database is not active), RACF uses global access checking tables, or a supplied profile, if any of these are present, to process resource access checking requests. (Note that RACF does not perform generic profile checking because a generic profile might allow access to a resource that an existing discrete profile already protects; had that profile been retrieved, RACF would not have allowed access to the resource.)

RACF calls RACROUTE REQUEST=AUTH and RACROUTE REQUEST=DEFINE preprocessing installation exits during failsoft processing. (RACF does not call the 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.

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, and no profile has been supplied, RACF calls the preprocessing installation exits. Failsoft processing then continues as follows:

- **RACROUTE REQUEST=AUTH:** RACF issues an operator intervention message to request permission to allow access to the resource.
- **RACROUTE REQUEST=DEFINE:** RACF issues an operator message to indicate that RACROUTE REQUEST=DEFINE has been issued and that the request is allowed.

  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 RDEFINE command to create a profile for the resource.

Note: Failsoft processing is not in effect when you deactivate RACF by using SETRACF INACTIVE or by modifying the ICHSECOP module.

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 resources for servicing. Also, they need the appropriate access authority to these resources.

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/VM: RACF Security Server Command Language Reference* for more information.)

Considerations for the RACF Database

As an alternative to maintaining all of your RACF profiles on one database, RACF allows you to have up to four RACF databases. The use of multiple RACF databases is recommended to reduce device contention and to reduce the number of resources made unavailable by the loss of one database or device. The optimum number of RACF databases for your installation depends on the extent to which you use RACF.

Note that the RACF databases should, themselves, be RACF-protected.

Protecting the RACF Database

The minidisks containing the RACF database should, themselves, be RACF-protected with minidisk profiles with UACC(NONE) and NOWARNING specified. The NOTIFY user ID should be the RACF security administrator. System programmers who may need to use the BLKUPD command to repair the RACF database must have UPDATE authority to the RACF database.

See *z/VM: RACF Security Server System Programmer's Guide* for details on how to maintain RACF databases, switch to alternate RACF databases, coordinate profile updates, and make other considerations when dealing with shared or multiple RACF databases.

Number of Resident Data Blocks

The use of resident data blocks reduces the number of I/O requests made to the RACF database. The default number of resident data blocks on z/VM is 100. If you need to change the number of resident data blocks, see *z/VM: RACF Security Server System Programmer's Guide* for details.

Using Dual Registration Panels

Note: When using RACF with DIRMAINT, the RACF dual registration panels should not be used. See *z/VM: Directory Maintenance Facility Tailoring and Administration Guide* for information on enabling DIRMAINT's RACF interface.

On z/VM systems, RACF provides dual registration panels so that you can add or delete users and minidisks in the RACF database and the z/VM directory at the same time. You can also use the dual registration panels to transfer a minidisk from one user to another. The dual registration menu, shown in Figure 21 on page 229, shows what you can do.

```
ICHP60                    DUAL REGISTRATION SERVICES
OPTION  ===>

Select one of the following options.

  1  ADD USER        -  Add a user to both the RACF database and the
                        z/VM directory.

  2  ADD MINIDISK    -  Add minidisks to an already defined user.


  3  DELETE USER     -  Delete a user from both the RACF database and the
                        z/VM directory.

  4  DELETE MINIDISK -  Delete minidisks from both the RACF database and the
                        z/VM directory.

  5  CHANGE MINIDISK -  Change the virtual address of a minidisk, transfer a
                        minidisk to another user, or both.

  6  FORMAT MINIDISK -  Format an already defined minidisk.


      PF01 = Help                           PF03 = End
```

Figure 21. Dual Registration Menu

To use dual registration panels, your installation must have ISPF Version 3 Release 2 or later and DirMaint Version 1 Release 5 or later.

When you install dual registration panels, you can specify default values for certain fields that appear on the ISPF dual registration panels by using the file DUALREG PROFILE. For example, you might want to protect all minidisks that you define through dual registration panels with a universal access authority (UACC) of NONE. You can specify a UACC of NONE for minidisks in DUALREG PROFILE, and this value will appear in the UACC field on all of the appropriate dual registration panels. (This is generally done during installation of RACF. For more information on installing RACF, see *Program Directory*.)

When using the panels, users can override the default values that appear on the ISPF dual registration panels. For example, suppose you have specified a default UACC of NONE for minidisks in DUALREG PROFILE and you want to define a system minidisk, SYSDISK.191, and give it a UACC of READ to allow all users to read it. When you access the appropriate dual registration panel to define SYSDISK.191, a value of NONE will appear in the UACC field. You can change the UACC to READ and this value will become the UACC for SYSDISK.191. However, the default UACC of NONE for minidisks specified in DUALREG PROFILE remains unchanged and will appear on the appropriate dual registration panel the next time you define a minidisk.

In order to use the dual registration panels, all of the following must be true:

1. You must be running with DirMaint command level 140A. If you are running with command level 150A, you can change your default command level to 140A with the following command before using the dual registration panels:

   ```
   DIRM DEFAULTS CMDLEVEL 140A
   ```

2. You must be authorized to issue the following DirMaint commands: ADD, AMDISK, CHVADDR, DIRECT, DMDISK, PURGE, and TMDISK in command level 140A for *all* user IDs. By default, these DirMaint commands are in command sets A and D. If running with ADD_COMMAND_PROCESSING=SHORT or with PURGE_COMMAND_PROCESSING=SHORT, the ADD and PURGE commands have probably been changed to command set S.
3. If you are adding user IDs with LINK statements in their directory entries, you also must be authorized for command set G for *all* users.

Note: The dual registration dialog does not support the use of the ACIGROUP control statement.

Using RACF with Other z/VM Products

DFSMS/VM

Data Facility Storage Management Subsystem for VM (DFSMS/VM) allows you to control your data and storage resources more efficiently. DFSMS/VM provides facilities for:

- Space management
- Minidisk management
- Interactive Storage Management Facility (ISMF)
- IBM Tape Library Dataserver Support

For more information, see "Authorizing DFSMS/VM User's" in *z/VM: DFSMS/VM Customization*, and "Preparing to Use Removable Media Services" in *z/VM: DFSMS/VM Removable Media Services*.

z/VM HCD

z/VM HCD enables you to define the hardware configuration using a graphical interface. This eliminates the need to use IOCP and RDEVICE statements, and ensures consistency of IOCDS and active I/O configuration through data validation at data-entry time, thus avoiding repeated IPLs. HCD may prove especially useful where Linux images are run as z/VM guests and familiarity with System z® I/O configuration is lacking. For more information, see "Installing HCM" in z/OS and z/VM: Hardware Configuration Manager User's Guide (https://www.ibm.com/docs/en/SSLTBW_2.5.0/pdf/eequ100_v2r5.pdf).

OSA/SF

The Open Systems Adapter Support Facility (OSA/SF) is a host-based tool supplied with z/VM that allows you to customize an OSA's modes of operation. You can access OSA/SF by a CMS user ID, by a REXX call to the OSA/SF API, or through a Java™-based graphical user interface (GUI). For more information, see Open Systems Adapter-Express Customer's Guide and Reference (https://www.ibm.com/docs/en/SSLTBW_2.3.0/pdf/ioa2z1f0.pdf).

For information about using z/VM TCP/IP with RACF, see Appendix A of *z/VM: TCP/IP Planning and Customization*

Chapter 15. LDAP Event Notification

LDAP event notification is used by IBM Tivoli Directory Integrator, in conjunction with password and password phrase envelopes (see Chapter 16, “Password and Password Phrase Enveloping,” on page 237), to enable a heterogeneous password synchronization solution.

RACF can be configured to create LDAP change log entries in response to changes in user, group, and general resource profiles. This support provides an open, remote method of change notification. Using only LDAP interfaces, an LDAP client can read the LDAP change log, detect updates to RACF users, groups, group membership, and general resources, and then retrieve RACF entries. To use this function, the LDAP server must be configured to enable the SDBM backend. For details, see Change logging in *z/VM: TCP/IP LDAP Administration Guide*.

Event notifications, through the creation of LDAP change log entries, are controlled by RACF resources in the RACFEVNT class. If RACFEVNT is active, and the appropriate resource is protected by either a discrete or generic profile, LDAP change log entries are created for the corresponding event types on a system-wide basis.

Table 41 on page 233 shows the name of the RACF resource in the RACFEVNT class used to control notifications for each type of supported change event.

Table 41. LDAP event notification of RACF profile changes

Resource in the RACFEVNT class	Change event type
`NOTIFY.LDAP.USER`	Password and password phrase changes, regardless of the command or method used
	Updates to a user's revoke status (that is, changes to the FLAG4 field in the USER profile), regardless of the command or method used
	Users added using the ADDUSER command
	User modifications made using the ALTUSER or PASSWORD command
	Users deleted using the DELUSER command
`NOTIFY.LDAP.GROUP`	Groups added using the ADDGROUP command
	Group modifications made using ALTGROUP command
	Groups deleted using the DELGROUP command
`NOTIFY.LDAP.CONNECT`	Group membership changes made using any of the following commands: • ALTUSER command, only when issued with GROUP, UACC, or AUTHORITY operand • CONNECT command • REMOVE command
	Users established in their default groups using the ADDUSER command
`NOTIFY.LDAP.`*`class-name`*	General resources added using the RDEFINE command
	General resource modifications made using the RALTER command
	Changes made using the PERMIT command to the standard or conditional access list of a general resource
	General resource deletions made using the RDELETE command

Note:

1. The RACF panels and the SDBM backend of LDAP issue RACF commands internally, so these interfaces are included.
2. The RACF panels can generate multiple commands while processing a user profile, and this might result in multiple change log entries.

LDAP Change Log Entries

The LDAP change log entry contains information such as the change initiator, the affected user, group, or general resource, the type of update (add, modify, or delete), and the time and date of the change. It does *not* contain a list of the RACF profile fields that were changed nor does it contain the new values for these fields.

In the case of a change to the standard or conditional access list of a general resource, the `changes` attribute of the change log entry indicates that a general resource profile was added, modified or deleted. The `changes` attribute does *not* identify the user or group permission that was added, modified, or removed.

In the case of a password or password phrase change, the `changes` attribute of the change log entry identifies the password or password phrase field as the changed field. The `changes` attribute does *not* contain the actual password or password phrase value but contains one of the following values:

`*ComeAndGetIt*`
: This indicates there is an encrypted password envelope or password phrase envelope that can be subsequently retrieved. (See Chapter 16, “Password and Password Phrase Enveloping,” on page 237 for details about envelopes.)

`*NoEnvelope*`
: This indicates there is no password envelope or password phrase envelope.

When other fields in a user's profile are changed in the same request that updates the values of the password and password phrase, three LDAP change log entries are created: one entry to log the password update, one to log the password phrase update, and another entry to log the information about the other changed fields. Removal of a user's password phrase does *not* create a separate log entry. (See Example 2.)

Example 1: An administrator issues the following command for a revoked user who is eligible for both password and password phrase enveloping.

```
ALTUSER userid PASSWORD(newpass) PHRASE(newphrase) RESUME OWNER(group-name)
```

If successful, this command causes *three* entries to be created to log the user profile changes.

- One entry identifies the password field as changed and contains the `*ComeAndGetIt*` value.
- A second entry identifies the password phrase field as changed and contains the `*ComeAndGetIt*` value.
- A third entry identifies changes in other non-specified fields; in this case, the change in the user's revoke status and owning group.

Example 2: An administrator issues the following command to update a user's password, remove the password phrase, and change the user's name.

```
ALTUSER userid PASSWORD(newpass) NOPHRASE NAME(new-user-name)
```

If successful, this command causes *two* entries to be created to log the user profile changes.

- One entry identifies the password field as changed and contains the `*ComeAndGetIt*` value.
- A second entry identifies changes in other non-specified fields; in this case, the change in the user's name and removal of the password phrase.

Example 3: An administrator issues the following command to remove a user's password phrase and change the user's name.

```
ALTUSER userid NOPHRASE NAME(new-user-name)
```

If successful, this command causes *one* entry to be created to log the user profile changes. This entry identifies changes in non-specified fields; in this case, the change in the user's name and removal of the password phrase.

See *z/VM: TCP/IP LDAP Administration Guide* for more information about the LDAP change log.

LDAP Change Log Queueing

If the LDAP server configured for change logging is unavailable at the time the RACF change occurs, then that change log entry will be queued, records will continue to be queued until a correctly configured LDAP server is started, or the number of queued records exceeds 256. Once the 256 maximum queue size threshold has been reached, new change log records will be discarded. The queueing of loss or change log records does not affect the RACF database itself; LDAP notification is attempted only after the RACF database has been updated. RACF will wait for an LDAP server configured for change logging to become available and automatically process the queue (without user intervention). If z/VM is restarted before the queue has been processed, all records on the queue will be lost.

Activating LDAP Change Notification

To activate LDAP change notification, define RACFEVNT class resources for the notifications you want and then activate the RACFEVNT class.

1. Define the RACFEVNT class resources for the LDAP notifications you want by creating one or more discrete profiles or by creating a generic profile.

   There are two ways you might activate *all* supported LDAP notification types: defining multiple discrete profiles or defining one generic profile. You can define a subset of resources based on the LDAP notifications you want.

   Example showing multiple discrete profiles:

   ```
   RDEFINE RACFEVNT NOTIFY.LDAP.USER
   RDEFINE RACFEVNT NOTIFY.LDAP.GROUP
   RDEFINE RACFEVNT NOTIFY.LDAP.CONNECT
   RDEFINE RACFEVNT NOTIFY.LDAP.FACILITY
   ```

   Example showing a generic profile:

   ```
   SETROPTS GENERIC(RACFEVNT)
   RDEFINE RACFEVNT NOTIFY.LDAP.*
   ```

2. Activate the RACFEVNT class and optionally RACLIST it to improve performance.

   ```
   SETROPTS CLASSACT(RACFEVNT) RACLIST(RACFEVNT)
   ```

3. Issue the following RACF commands to complete the setup:

   ```
   RDEFINE FACILITY IRR.CHANGELOG UACC(NONE)
   PERMIT IRR.CHANGELOG CLASS(FACILITY) ID(LDAPSRV) ACCESS(READ)
   SETROPTS RACLIST(FACILITY) REFRESH
   ```

Restrictions on refreshing SETROPTS and VMXEVENT

SETROPTS

Because there might be multiple profile updates before a refresh activates changes, automatic refreshing of SETROPTS options is not implemented. Instead, refreshing is under the control of the user. New SDBM

entries embody the RACF SETROPTS commands. Modifying these entries results in issuing a SETROPTS command. For example, if the modification contains:

```
racfraclist: facility
racfsetroptsattributes: refresh
```

the SDBM issues:

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

A limitation is that the SDBM can issue the SETROPTS REFRESH command only on the RACF service machine to which LDAP is connected. This service machine is the one identified in the RACF SERVMACH file to receive RACROUTE requests (the default service machine is RACFVM). In a multi-server environment, service machines other than the RACF require manual intervention to implement SETROPTS changes (refresh in-storage profiles, and so forth).

VMXEVENT

While it is possible for the SDBM backend to issue the command SETROPTS RACLIST(FACILITY) REFRESH to refresh SETROPTS options, it is currently not possible for SDBM to issue the command SETEVENT REFRESH to activate VMXEVENT profile changes.

Chapter 16. Password and Password Phrase Enveloping

This topic provides information about creating *password envelopes* and *password phrase envelopes* to enable authorized applications to recover user passwords and password phrases. Envelopes are used by IBM Tivoli Directory Integrator, in conjunction with LDAP notification (see Chapter 15, "LDAP Event Notification," on page 233), to enable a heterogeneous password synchronization solution.

Overview of Enveloping

RACF can be configured to save user passwords and password phrases so that an authorized application can recover them in clear text. This ability can be restricted to a subset of your users and can be further limited to only passwords or password phrases.

When an eligible user's password or password phrase is changed, the new value is encrypted under a public key within a key database file associated with the LDAP server configured for change logging. The encrypted value is then stored in the user's profile. When an application requests the password or password phrase, RACF decrypts the value, and then encrypts it in PKCS #7 format for recipients whose digital certificates have been placed in the same key database file. An authorized application can then decrypt the *password envelope* or *password phrase envelope* using the recipient's private key.

Generally, any new password or password phrase is enveloped for an eligible user, with the following exceptions:

- Initial ADDUSER passwords and password phrases.
- When the new value of the password or password phrase is the same as the current value.
- When an application uses RACROUTE or ICHEINTY, rather than a RACF command, to set the password, and the password contains characters that are not accepted by the RACF commands.
- When an application uses RACROUTE or ICHEINTY to set the password and specifies ENCRYPT=NO.
- When an application uses ICHEINTY to set the password phrase but the password phrase does not have a valid (9–100 character) length.

Resources that Control Enveloping

The PASSWORD.ENVELOPE and PASSPHRASE.ENVELOPE resources in the RACFEVNT class control whether new passwords and password phrases are enveloped for a given user. You can optionally control both password and password phrase enveloping using a single generic profile, such as PASS*.ENVELOPE. If the user whose password or password phrase is changed has at least READ access to the appropriate resource, then the new password or password phrase is enveloped. Thus, you can use these resources to selectively authorize users whose passwords or password phrases will be enveloped. For example, you can exclude sensitive user IDs from both password and password phrase enveloping by authorizing those IDs to the PASS*.ENVELOPE resource with access level NONE.

Restrictions:

- An enveloped password or password phrase is not displayed in the user's LISTUSER output. (The lines `PASSWORD  ENVELOPED=YES` and `PHRASE  ENVELOPED=YES` in the LISTUSER output indicates when a password or password phrase envelope is present. See *z/VM: RACF Security Server Command Language Reference* for LISTUSER details.)
- An enveloped password or password phrase is not unloaded by the database unload (IRRDBU00) utility. (Certain fields in the output indicate that a password or password phrase envelope is present. See *z/VM: RACF Security Server Macros and Interfaces* for details about IRRDBU00 output records.)

- No audit records are created as a result of failed access checks to resources in the RACFEVNT class. You can set audit options in the resource profiles to log successes, and thus maintain a history of whose passwords and password phrases are enveloped.
- If the user fails verification (when the RACROUTE REQUEST=VERIFY is executed during envelope processing), the user's new password or password phrase is *not* enveloped, even when the password or password phrase change is successful. One possible reason for a verification failure (during envelope processing) is that the user is *revoked* at the time that envelope processing occurs.

  For example, if an administrator uses the ALTUSER command to change the password of a revoked user who is eligible for password enveloping, the user's password is changed but the user's password is *not* enveloped. Even when the administrator subsequently resumes the revoked user, the password is *not* enveloped.

  To envelope the password or password phrase of an eligible user who is revoked, you must resume the user before the change, or resume the user with the same ALTUSER command that changes the password or password phrase.

  Example (correct):

  ```
  ALTUSER userid PASSWORD(new-password) PHRASE(new-password-phrase) RESUME
  ```

  Example (correct):

  ```
  ALTUSER userid RESUME
  ALTUSER userid PASSWORD(new-password) PHRASE(new-password-phrase)
  ```

  Example (incorrect):

  ```
  ALTUSER userid PASSWORD(new-password) PHRASE(new-password-phrase)
  ALTUSER userid RESUME
  ```

  When you use the correct examples, the revoked user's new password and password phrase are enveloped.

Signing Hash Algorithm and Encryption Strength Used to Create the Envelope

Both the signing hash algorithm and encryption strength are configurable attributes. Use application data (APPLDATA) in the RACFEVNT resource profiles to specify the signing hash algorithm that signs the PKCS #7 envelope, and the encryption strength used when encrypting the envelope. The syntax of the APPLDATA string consists of a character string indicating the signing hash algorithm, followed by a forward slash (/), followed by a string indicating the encryption strength.

Examples:

```
RDEFINE RACFEVNT PASSPHRASE.ENVELOPE UACC(NONE) APPLDATA('MD5/STRONG')
RALTER  RACFEVNT PASSWORD.ENVELOPE APPLDATA('MD5/STRONG')
```

Allowable values for the signing hash algorithm:

- MD5 (default)
- SHA1

Allowable values for the encryption strength:

- STRONG (default)
- MEDIUM
- WEAK

Guideline: Use the strongest encryption possible. If you must use weaker encryption, protect yourself against offline attacks by carefully controlling access to the RACF database and any other repository where envelopes may be stored after being retrieved from RACF.

Encryption strength value	Data encryption method
STRONG	Triple DES (a 168-bit encryption key)
MEDIUM	DES (a 56-bit encryption key)
WEAK	RC2 (a 40-bit encryption key)

If the APPLDATA is not specified in the profile, the defaults are taken as noted above. If an empty qualifier exists in the APPLDATA, then the default value is used for that qualifier. For example, if the APPLDATA is specified as SHA1, then SHA1 is used as the signing hash algorithm, and triple DES is used as the encryption algorithm. If the APPLDATA is specified as `/MEDIUM`, then MD5 is used as the signing hash algorithm, and DES is used as the encryption algorithm.

If the APPLDATA is specified incorrectly, an error message is issued to the console. Thereafter, the default values are used whenever users who are eligible for enveloping change their passwords or password phrases, or whenever an application requests the retrieval of an envelope.

The APPLDATA can be changed at any time.

The Change Logging SSL Key Database (KDB)

The SSL KDB is used to contain RACF and recipient certificates. This key database will be used for enveloping both passwords and password phrases.

The SSL Key Database is a KDB which is associated with the LDAP server configured for change logging. It contains certificates of all the principals who are intended to retrieve a user's changed password from RACF. A changed password is encrypted using the public keys contained within these certificates. It also contains a certificate and private key for the RACF server itself. New passwords are signed using the private key of the RACF server. The RACF server is also treated as a recipient of the password change. RACF will encrypt passwords for up to 20 certificates in this KDB (to maintain consistency with z/OS RACF behaviour).

Controlling Envelope Retrieval

The KDB consists of two files the kdb file itself and an associated password stash file. These two files by default reside in the home directory of the LDAP server configured for change logging, with the names IRR.PWENV.KEYRING and IRR.PWENV.KEYRING.sth respectively. If alternative file names or locations are used then the pathnames must be specified in the LDAP configuration file (DS ENVVARS). See Configuring the LDAP Server, Step 9. Set Environment Variables in *z/VM: TCP/IP Planning and Customization* for more information. The variable names are IRR.PWENV.KEYFILE and IRR.PWENV.PWSTASH and they can specify either an absolute pathname or a pathname relative to the LDAP server's home directory.

An LDAP server must have update access to the ICHCONN profile in the FACILITY class to retrieve envelopes. In addition, the requesting userid must have access to the appropriate resource in the FACILITY class.

The following resources in the FACILITY class control the retrieval of envelopes from userids making retrieval requests via the LDAP SDBM backend.

Resource name	Controls retrieval of ...
IRR.RADMIN.EXTRACT.PWENV	Only password envelopes
IRR.RADMIN.EXTRACT.PPENV	Only password phrase envelopes
IRR.RADMIN.EXTRACT.*	Both password and password phrase envelopes

You can set the audit options for these resources to log successes, and thus maintain a history of whose passwords and password phrases are retrieved, and by whom. Failures can also be logged. (The log string identifies the user whose password or password phrase was retrieved.)

The NOTIFY.LDAP.USER Resource

When a resource is defined in the RACFEVNT class called NOTIFY.LDAP.USER, an LDAP change log entry is created when a user's password or password phrase is changed. A change log entry is created for user additions, modifications, and deletions if this resource is protected (by either a discrete or generic profile). No authorization checks are driven against this resource.

Setting up Enveloping

There are several RACF setup steps to perform in order for recipients to be able to retrieve user password and password phrase changes. See the setup steps in the following topics:

1. "Defining a Local Certificate Authority Certificate Acting as the Certificate Authority" on page 241
2. "Generating an X.509 V3 Certificate for RACF's Use for Enveloping" on page 242
3. "Certificate Management for RACF Change Logging and Enveloping" on page 240
4. "Authorizing the Envelope Recipient" on page 245
5. "Activating LDAP Change Notification and Enveloping" on page 246.

Tip: While you are initially configuring and testing this function, check the console for error messages. As RACF detects errors during the enveloping process, they will be reported to the console, not to the end user who initiates a password or password phrase change.

Certificate Management for RACF Change Logging and Enveloping

This section is based on the following assumptions :

- The SSL utility program gskkyman will be used for certificate management.
- The userid running gskkyman has write access to the LDAP server's BFS home directory, or is able to copy the key database there upon completion.
- The userid running gskkyman has linked and accessed TCPMAINT's 591 and 592 disks where SSL code has been installed.
- The intended recipient of a password or password phrase envelope has a RACF user ID and password or password phrase that it uses to bind to the LDAP server. This user ID is also used to authorize the retrieval of envelopes using a resource in the FACILITY class. In the following examples, the user ID used is ITDI.
- The key database name is IRR.PWENV.KEYRING and the stash file name is IRR.PWENV.KEYRING.sth. These are the default names used by RACF. If other names are used, you must set the environment variables IRR.PWENV.KEYFILE and IRR.PWENV.PWSTASH in the LDAP configuration file DS ENVVARS to the pathnames used, which can be either an absolute pathname or a pathname relative to the LDAP server's home directory.

  For further information on gskkyman, see *z/VM: TCP/IP User's Guide*.

  Note: If a value is set for IRR.PWENV.KEYFILE but not for IRR.PWENV.PWSTASH, the pathname of the password stash file will be derived from the key database pathname so that if the kdb pathname has an extension of three characters (for example, .kdb), it is replaced with .sth. Otherwise, .sth is simply appended.

Defining the Change Logging and Enveloping SSL Key Database

Note: The following gskkyman dialogs are samples only and appropriate values should be used to reflect your installation standards and security requirements.

1. Enter gskkyman.
2. Select `Option 1 Create new database`.
3. Enter `IRR.PWENV.KEYRING` at the `Enter key database name` prompt.

4. Enter appropriate values for password, password expiration, and database record length prompts or take defaults if available.
5. You should receive the following message :

```
Key database database name created.
Press ENTER to continue.
```

6. Enter 0 to exit or press ENTER to continue as below (after Open database step).

Store Database Password for Use by LDAP Server

1. Enter gskkyman.
2. Select Option 2 Open database.
3. Enter IRR.PWENV.KEYRING at the Enter key database name prompt.
4. Enter password.
5. Select Option 10 Store database password.
6. You should receive the following message :

```
Database password stored in database name.sth
Press ENTER to continue.
```

7. Enter 0 to exit or press ENTER to continue as below (after Open database step).

Defining a Local Certificate Authority Certificate Acting as the Certificate Authority

If you will be using RACF as your certificate authority (CA), you will need to create a certificate authority certificate, if you have not already done so. The following gskkyman dialog creates a certificate authority (or signer) certificate. This example creates a certificate called RACFCA to be used when creating subsequent certificates, such as the certificate which RACF will use during the password enveloping process. The input may be modified, specifically subjectsdn, organization, and country, to reflect the naming structure and conventions used at your installation.

1. Enter gskkyman.
2. Select Option 2 Open database.
3. Enter IRR.PWENV.KEYRING at the Enter key database name prompt.
4. Enter password.
5. Select Option 6 Create a self-signed certificate.
6. Select Option 1 CA certificate with 1024-bit RSA key.
7. Select Option 1 SHA-1.
8. Enter RACFCA at the Enter label prompt.
9. Enter RACFCA at the Common name prompt.
10. Enter appropriate values for Organizational unit, Organization, City/Locality, State/Province, Country/Region, and Number of days valid prompts or take defaults if available.
11. Enter Option 0 at the Enter 1 to specify subject alternate names or 0 to continue: prompt.
12. You should receive the following message :

```
Certificate created.
Press ENTER to continue.
```

13. Enter 0 to exit or press ENTER to continue as below (after Open database step).

Note: Certificates signed with this local certificate authority certificate show an issuer of cn=RACFCA,o=*ooo*,c=*cc* (where *ooo* and *cc* are values entered above) when listed. RACF will sign the

password envelope so that a recipient can verify that the envelope signature was created using RACF's certificate (which is created in the next step). If the recipient also wants to check the veracity of RACF's certificate, it will need this CA certificate to do so. In this case, the CA certificate must be exported to a keyring known to the recipient application and marked as trusted.

Generating an X.509 V3 Certificate for RACF's Use for Enveloping

A digital certificate containing a private key must be generated for RACF's use. This certificate and its associated private key will be used in the RACF password enveloping process. In the following example, gskkyman is used to generate a certificate for the RACF server, whose RACF userid is RACFVM. This certificate is signed by RACFCA, which is the certificate authority in the context of this example. The local CA certificate is identified by the label RACFCA.

1. Enter gskkyman.
2. Select `Option 2 Open database`.
3. Enter `IRR.PWENV.KEYRING` at the `Enter key database name` prompt.
4. Enter password.
5. Select `Option 1 Manage keys and certificates`.
6. Select RACFCA certificate.
7. Select `Option 10 Create a signed certificate and key`.
8. Select `Option 5 User or server certificate with 1024-bit RSA key`.
9. Enter RACFVM at the `Enter label` prompt.
10. Enter RACFVM at the `Common name` prompt.
11. Enter appropriate values for `Organizational unit`, `Organization`, `City/Locality`, `State/Province`, `Country/Region`, and `Number of days valid` prompts or take defaults if available.
12. Enter `Option 0` at the `Enter 1 to specify subject alternate names or 0 to continue:` prompt.
13. You should receive the following message :

    ```
    Certificate created.
    Press ENTER to continue.
    ```

14. Enter 0 to exit or press ENTER to continue as below (after Open database step).

Make the RACFVM certificate the default certificate in the key database (KDB). RACF's certificate must be defined as the default certificate:

1. Enter gskkyman.
2. Select `Option 2 Open database`.
3. Enter `IRR.PWENV.KEYRING` at the `Enter key database name` prompt.
4. Enter password.
5. Select `Option 1 Manage keys and certificates`.
6. Select RACFVM certificate.
7. Select `Option 3 Set key as default`.
8. You should receive the following message :

    ```
    Default key set.
    Press ENTER to continue.
    ```

9. Enter 0 to exit or press ENTER to continue as below (after Open database step).

Using these examples will create a certificate which is TRUSTED. However, it's always a good idea to verify this by listing the certificate using `option 1` of the `Key and Certificate Menu`. The trust indicator can be set using option 4 of this menu. This also applies to the certificates which are created in the following steps.

At the time a user's password is changed, if the user is eligible for enveloping then the user's new password is encrypted under the public key of the default certificate only, and stored back in the user's USER profile. When an application requests the envelope, the password is decrypted using the private key of the default certificate, and a PKCS#7 password envelope is created for all of the recipients in the key database.

Note: Any user with READ access to the key database may be able to obtain the default certificate's private key and thus any user's encrypted password. As always, you should protect the KDB and its copies, as well as the RACF database and its copies, against inappropriate access. Further, you should verify that applications retrieving passwords do so using the LDAP server.

Generating an X.509 V3 Certificate for the Password Envelope Recipient

During the RACF password enveloping process, RACF encrypts data which can only be recovered by the intended recipient of that data. An intended recipient, such as the identity of the ITDI process running off of the z/VM platform, is identified by a X.509 V3 certificate. Certificates which identify intended recipients of RACF password envelopes must be loaded in to the KDB. Note that these certificates are only used to encrypt password information; they are not used to bind to LDAP or to authenticate to RACF. Generally speaking, you should only permit trusted application identities (not humans) to recover user passwords. Certificates for intended recipients may be created and exported to off platform processes for instance. The creation of the certificates may be accomplished using the following gskkyman dialog which generates a certificate for ITDI, who in this example is the identity of the process which will be authorized to retrieve RACF password envelopes. This certificate is signed with the local CA certificate that is identified by the label RACFCA.

1. Enter gskkyman.
2. Select `Option 2 Open database`.
3. Enter `IRR.PWENV.KEYRING` at the `Enter key database name` prompt.
4. Enter password.
5. Select `Option 1 Manage keys and certificates`.
6. Select RACFCA certificate.
7. Select `Option 10 Create a signed certificate and key`.
8. Select `Option 5 User or server certificate with 1024-bit RSA key`.
9. Enter `recipient_ITDI` at the `Enter label` prompt.

   Note: For keys that do not have a label starting with "recipient_" the key management tool that generates the recipient certificate must support key usage extensions, specifically the recipient keys are selected by the inclusion of the keyEncipherment usage extension. GSKKYMAN does support these usage extensions whereas keytool does not.
10. Enter `ITDI` at the `Common name` prompt.
11. Enter appropriate values for `Organizational unit`, `Organization`, `City/Locality`, `State/Province`, `Country/Region`, and `Number of days valid` prompts or take defaults if available.
12. Enter `Option 0` at the `Enter 1 to specify subject alternate names or 0 to continue:` prompt.
13. You should receive the following message :

    ```
    Certificate created.
    Press ENTER to continue.
    ```
14. Enter 0 to exit or repeat the above steps to create more user certificates

Note: RACF will encrypt the password for no more than 20 recipient certificates.

You may also create certificates directly on the recipient platform and import them into the key database file. This has the advantage of keeping the private key only where it is needed. Any key management system can be used to create the recipient key pair and certificate, as long as it can export certificates in an industry standard format understood by the gskkyman utility.

The following example uses keytool, the key management tool available with many Java virtual machines (JVMs), including the JVM shipped with IBM Tivoli Directory Integrator. You may not need to perform all of the steps below if you already have a key management infrastructure. The examples below assume you are starting new.

Creating the Certificate

Use of RSA as a key algorithm is required. (If keytool uses the default value DSA, you must change it to RSA).

```
keytool -genkey -alias ITDI -keypass xxxxxx
  -storepass xxxxxx -keystore recipient.jks
  -dname "CN=(IBM Directory Integrator Server 1),O=(ibm),C=(us)"
  -keyalg RSA -validity 1825
```

The keytool command above creates a self-signed certificate which expires in five years. For example purposes, this is sufficient. In a production environment, you may wish to use something other than a self-signed certificate.

Exporting the Certificate

The rfc keyword specifies base64-encoded output.

```
keytool -export -alias ITDI -file ITDI.b64 -keystore recipient.jks -storepass
  xxxxxx -rfc
```

Omitting the rfc keyword will export the certificate in binary format.

Copying the Certificate to the Host System

Because the certificates were exported in base64, they can be cut and pasted into a host file using a text editor. If you use FTP, transfer them using in ASCII (not binary) mode. The files should start with -----BEGIN CERTIFICATE----- and end with -----END CERTIFICATE----- when viewed on the host. If exported in binary format, FTP should be used in binary mode.

For this example, the file is copied to ITDI.CERT in the BFS home directory of the LDAP server.

1. Enter gskkyman.
2. Select Option 2 Open database.
3. Enter IRR.PWENV.KEYRING at the Enter key database name prompt.
4. Enter password.
5. Select Option 7 Import a certificate.
6. Enter ITDI.CERT at the Enter import file name prompt.
7. Enter recipient_ITDI at the Enter label prompt.

   Note: If the certificate was generated by a key management tool that does not support keyEncipherment usage extensions, such as keytool, then the label *must* begin with the string "recipient_" in order for it to be recognized as a recipient.
8. You should receive the following message:

   ```
   Certificate imported.
   Press ENTER to continue.
   ```
9. Enter 0 to exit or press ENTER to continue.

RACF constructs the PKCS#7 password envelope at the time the envelope is requested. So, if you add a certificate for a principal, that principal will be able to decrypt the password for any eligible user whose current password has already been changed (assuming the principal has the authorization which is described in the next step). Likewise, if a certificate is removed from the KDB, the principal will not be

able to decrypt any password envelopes, even if the password was changed when the certificate was in the KDB, and even if the authorization described in the next step has not been removed for the principal.

Exporting RACF's Certificate to the Recipient Key Database

If you have implemented IBM Tivoli Directory Integrator, or the recipient application intends to verify the signature of the password envelopes as they are decrypted to ensure they are from RACF, then the RACF (signing) certificate must be available to the recipient in the recipient key database. If the application validates the entire trust chain of the signing certificate, then the CA certificate chain is also required (in the example above, there is only a single CA certificate). ITDI does not require the trust chain, but does require the signing certificate. The following examples show how to export the RACF certificate. Repeat and alter the steps as appropriate for the CA certificate(s). Export the RACF certificate. (This certificate was created in “Generating an X.509 V3 Certificate for RACF's Use for Enveloping” on page 242.

1. Enter gskkyman.
2. Select Option 2 Open database.
3. Enter IRR.PWENV.KEYRING at the Enter key database name prompt.
4. Enter password.
5. Select Option 1 Manage keys and certificates.
6. Select RACFVM certificate.
7. Select Option 6 Export certificate to a file.
8. Select Option 2 Base64 ASN.1 DER in the Export File Format menu
9. Enter racfvm.b64 at the Enter export file name prompt.
10. You should receive the following message:

```
Certificate exported.
Press ENTER to continue.
```

11. Enter 0 to exit or press ENTER to continue

This file must be transferred to the recipient system. Use FTP (ASCII mode) or simply cut and paste it to create the racfvm.b64 file. Then, import the file using the keytool command:

```
keytool -import -alias RACFVM -file racfvm.b64 -keystore recipient.jks -storepass
xxxxxx
```

Authorizing the Envelope Recipient

Authorize these same principals to the new extract envelope function (to retrieve envelopes from RACF) using one of the following examples. Example 1 allows you to separately control retrieval of password envelopes and password phrase envelopes. Example 2 allows you to control retrieval of both password envelopes and password phrase envelopes using the same resource.

The FACILITY resource names shown in these examples are described in “Controlling Envelope Retrieval” on page 239.

Example 1:

```
RDEFINE FACILITY IRR.RADMIN.EXTRACT.PWENV UACC(NONE)
PERMIT IRR.RADMIN.EXTRACT.PWENV CLASS(FACILITY) ID(ITDI) ACCESS(READ)

RDEFINE FACILITY IRR.RADMIN.EXTRACT.PPENV UACC(NONE)
PERMIT IRR.RADMIN.EXTRACT.PPENV CLASS(FACILITY) ID(ITDI) ACCESS(READ)
```

Example 2:

```
RDEFINE FACILITY IRR.RADMIN.EXTRACT.* UACC(NONE)
PERMIT IRR.RADMIN.EXTRACT.* CLASS(FACILITY) ID(ITDI) ACCESS(READ)
```

Guideline: In general, authorize only trusted *applications*, not *users*, to extract envelopes.

Failed access attempts to these resources are logged by default. The log string of the audit record contains the user ID whose envelope is being retrieved. If you use a generic profile (shown in Example 2), look for the resource name in the audit record and you can distinguish whether a password envelope or password phrase envelope was retrieved.

Guideline: Given the sensitive nature of this function, you should log successful accesses as well. For example, a user with the RACF AUDITOR attribute can execute the following command:

```
RALTER FACILITY IRR.RADMIN.EXTRACT.* GLOBALAUDIT(ALL(READ))
```

If the FACILITY class is already ACTIVE and RACLISTed, refresh the class.

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

Activating LDAP Change Notification and Enveloping

This procedure involves defining resources in the RACFEVNT class, and activating this class.

1. Define the profile which controls password enveloping. This profile is called PASSWORD.ENVELOPE. The APPLDATA of this profile is used to specify the signing hash algorithm used to sign the PCKS#7 password envelope, and the encryption strength used when encrypting the password envelope. For example, to specify the strongest signing and encryption:

   ```
   RDEFINE RACFEVNT PASSWORD.ENVELOPE UACC(NONE) APPLDATA('MD5/STRONG')
   ```

2. Enable password enveloping for users whose passwords are to be encrypted for the intended recipients whose digital certificates were set up above. This is done by permitting a given user or group to PASSWORD.ENVELOPE:

   ```
   PERMIT PASSWORD.ENVELOPE CLASS(RACFEVNT) ID(USER1 USER2 GROUPA GROUPB)
    ACCESS(READ)
   ```

3. The profile may be given UACC(READ) to activate system-wide password propagation, or specific users and groups may be permitted. However, assigning a UACC(READ) to the PASSWORD.ENVELOPE profile will also enable the enveloping of passwords for highly privileged users (i.e. system SPECIAL users, or users which are defined for emergency recovery purposes). This approach is not recommended, unless sensitive user IDs are specifically permitted to this resource with ACCESS(NONE). Ultimately of course, this is up to the policy of the individual customer. If a non-global approach is taken, keep in mind that over time, one must consider how newly added users and groups fit into the access scheme, so that users' passwords are properly enveloped or not. In particular, it may be important to have the user's intended group connections well in place before the user performs his initial logon in which he needs to change his password. Also keep in mind that if a user is connected to several groups, their effective authority is the highest allowed by any of their groups (assuming they aren't specifically permitted by user ID, in which case this authority overrides that granted by any of the groups). For example, if list-of-groups processing (SETROPTS GRPLIST)is active, and user BOB is connected to groups GROUP1 and GROUP2, and GROUP1 is permitted with ACCESS(NONE), and GROUP2 is permitted with ACCESS(READ), and BOB is not explicitly permitted, then BOB's effective access to PASSWORD.ENVELOPE is READ, and BOB's password will be enveloped.

   Note: RACFEVNT profile PASSPHRASE.ENVELOPE is used to control password phrase enveloping, in the same way as PASSWORD.ENVELOPE controls password enveloping. A generic profile (PASS*.ENVELOPE) can be used to provide common control of the password and password phrase enveloping functions, as follows:

   ```
   SETROPTS GENERIC(RACFEVNT)
   RDEFINE RACFEVNT PASS .ENVELOPE UACC(NONE) APPLDATA('MD5/STRONG')
   ```

   There is also the option of maintaining separate discrete profiles. This gives the flexibility to envelope just passwords, or just password phrases. It also provides the ability to make different sets of users eligible for password enveloping vs. password phrase enveloping.

4. Optionally, an LDAP change log entry can be created whenever a user's new password is enveloped. This step is required if you use an application like IBM Directory Integrator to implement a heterogeneous password synchronization solution. This is done by defining the new RACFEVNT profile named NOTIFY.LDAP.USER:

   ```
   RDEFINE RACFEVNT NOTIFY.LDAP.USER UACC(READ)
   ```

   A generic profile could also be used, as follows:

   ```
   SETROPTS GENERIC(RACFEVNT)
   RDEFINE RACFEVNT NOTIFY.LDAP.
   ```

5. Activate the functions by activating the RACFEVNT class. It can be RACLISTed to improve performance, but this is not required.:

   ```
   SETROPTS CLASSACT(RACFEVNT) RACLIST(RACFEVNT)
   ```

Planning Considerations for Heterogeneous Password Synchronization

Before implementing enveloping, carefully weigh the risks against the benefits. RACF has always implemented one-way encryption when storing new user passwords and password phrases. This implementation makes it impossible for even a system administrator to obtain a user's password or password phrase once that user has changed the initial logon value. This implementation protects users against unauthorized use of their passwords and password phrases, and increases system accountability. Implementing the enveloping function makes it possible for an authorized user or application to retrieve a user's current password and password phrase. Subsequent use of this password or password phrase will result in a loss of accountability, resulting in the question: *Who actually entered the user ID and password or password phrase and is now working under the user's identity?* A RACF administrator currently has the capability of simply changing the user's password or password phrase, and then logging on as that user. However, when this occurs, the user will become aware at next logon time that his password or password phrase was changed.

There are other solutions that perform password synchronization. Such applications use RACF exits to intercept password changes. The PKCS #7 enveloping function, in conjunction with LDAP event notification, provides a simpler way for such applications to subscribe to password and password phrase changes, but does not by itself provide a higher or lower degree of security than is already put in place by such applications. Ultimately, you must rely on the application to maintain the security of RACF passwords and password phrases from the point those values are intercepted. For example, the application should not send a password or password phrase across a network in clear text and should protect any repository that might contain these values in clear text.

It is the installation's choice to evaluate password synchronization software, and enable PKCS #7 enveloping in support of such software. Part of deploying such software is ensuring proper user education and network security. Several RACF implementation features help to minimize the risks:

- As with all new RACF enhancements, enveloping is not enabled by default. You must enable it before any passwords or password phrases are enveloped and retrievable.
- Public key cryptography protects the envelopes as they are sent across the network, and makes them available only to authorized principals whose certificates you connect to a RACF key database file. Using RACF key database files to authorize principals keeps the trust policy within your scope, as the RACF security administrator.
- The encryption and retrieval of envelopes is fully dynamic with respect to key database file changes. (The envelope is created on demand when it is requested, using the current contents of the key database file.) Therefore, if the private key of a recipient is compromised, you can remove the recipient's certificate from the RACF key database file to dynamically render envelopes indecipherable to the thief of the recipient's private key. Even if the thief is able to masquerade as the intended recipient, bind to LDAP, and retrieve an envelope under the recipient's identity, he will not be able to decipher it using the stolen private key.

- Profiles in the RACFEVNT class establish your enveloping policy. This policy allows you to scope password and password phrase enveloping to a subset of RACF users, allowing you to exclude sensitive user IDs, and control enveloping for passwords and password phrases independently, at your discretion.
- Policy changes are audited for each privileged operation associated with enveloping, such as changes to the enveloping policy, and actual retrievals of envelopes from RACF.

Chapter 17. Security Classification and Zoning

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 corresponding to a department or an area within an organization in which the users have similar security requirements. A *security label (SECLABEL)* is an installation-defined name corresponding to a security level and zero or more security categories.

While the functions described in this chapter may be useful in some environments, their primary intent is to implement a system which conforms to the requirements of the Common Criteria. For more details on configuring such a system, see *z/VM: Secure Configuration Guide*.

This section discusses security levels and security categories first. Security labels are discussed later (see "Understanding Security Labels" on page 253).

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 protected by a profile that has security category and/or security level data. (For information on global access checking, see Chapter 7, "Fast Authorization Using the Global Access Checking (GAC) Table," on page 121. For a complete list of the sequence of checks that RACF makes to grant or deny access to a resource, see "Authorization Checking for Resources Protected by RACF Profiles" on page 309.)

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 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). See "Authorization Checking for Resources Protected by RACF Profiles" on page 309.

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 and/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 313.

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 will 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 class-name profile-name SECLEVEL(security-level-name)

RALTER class-name 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 and/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, z/OS 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. (These commands are fully documented in *z/VM: RACF Security Server Command Language Reference*.) The examples assume that the SECLEVEL and CATEGORY tables shown earlier have been defined.

Examples

1. Protect a z/VM minidisk with security category and security level information.

```
RDEFINE VMMDISK SMITH.191 SECLEVEL(ROUTINE)
        ADDCATEGORY(ACCOUNTING)
```

This command creates a general resource profile with a security level of ROUTINE and a security category of ACCOUNTING.

2. Modify the security level information and add a category in a general resource profile for a z/VM minidisk.

```
RALTER VMMDISK SMITH.191 SECLEVEL(CONFIDENTIAL)
       ADDCATEGORY(PERSONNEL)
```

This command modifies the profile to have a security level of CONFIDENTIAL instead of ROUTINE and adds a security category of PERSONNEL.

The security classification information in a user's profile is an access allowance, while 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 will 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.

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 (for example, done with the RLIST command) will show an UNKNOWN category. To delete this category, enter a command like the following (with no category specified on the DELCATEGORY operand):

```
RALTER  class-name  profile-name  DELCATEGORY
```

To search for such profiles, enter the search command as follows:

```
SEARCH  CLASS(class-name)  CATEGORY
```

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 can have a PSF identification label related to the security label of the user or job printed on every page.
- Output printed for a user by the z/VM system or by RSCS Secure Printing can have an identification label related to the security label of the user printed for each output file.

- 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).

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(s) 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 256. If these instructions are not followed, then the resource will become inaccessible. Depending on the nature of the resource involved, the system could become unusable. To recover from such a situation, log on as a user with the system-SPECIAL attribute, specifying SYSHIGH as the current security label. Assign security labels or issue SETROPTS NOCLASSACT(SECLABEL).
- 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 "Requiring Security Labels" on page 260.
- 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 313.
- 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 Users from Copying Data from One Security Label to a Lower Security Label" on page 259.
- If your system includes products that do not support security labels when they invoke RACF, you should consider using the SETROPTS COMPATMODE option. See "Compatibility Mode for Security Labels" on page 260.

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.

Note:

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/VM: RACF Security Server Command Language Reference*. For z/VM considerations when the security label class is active, see “Activating the Security Label Class on z/VM” on page 261.

Security Classification of Printed Output

When mandatory access controls (security labels) are active, CP can add a security classification to the separator pages generated on printers managed by CP. For more information, see "CP Printer Support" in *z/VM: Secure Configuration Guide*, and the SECTABLE utility in *z/VM: CP Commands and Utilities 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 will be ignored. Even though SECLEVEL and CATEGORY information will be ignored, you can maintain it (for example, using RACF commands). If the SECLABEL class is not active, then RACF will continue to use the SECLEVEL and CATEGORY function as it is currently implemented. Thus by activating or deactivating the SECLABEL class using the SETROPTS command, an installation can 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 that you need not, and indeed cannot, create directly. (At 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 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, do the following two commands:

```
RLIST  SECDATA  SECLEVEL

RLIST  SECDATA  CATEGORY
```

Figure 22 on page 256 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).

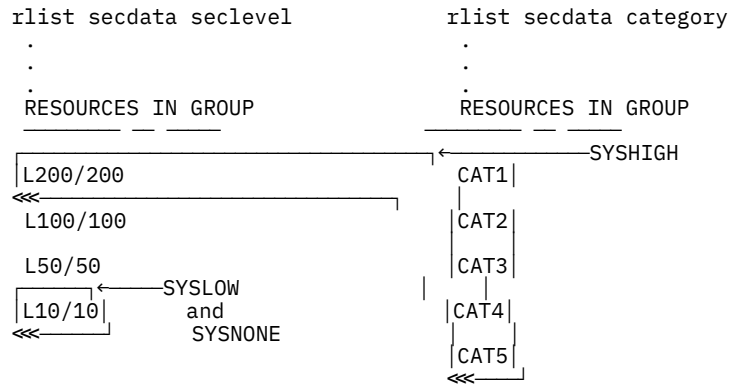


Figure 22. Sample SYSHIGH and SYSLOW Security Labels

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.

Security Label NONE

You should not define a security label on z/VM with the name NONE for following reasons:

- NONE is the default security label for z/VM system printers.
- VM does not permit jobs to print on a printer with a security label of NONE.
- When a spool file has no security label, the response to a QUERY READER, QUERY PRINTER, QUERY PUNCH or QUERY TRFILES command will contain the character string NONE in the SECLABEL field.

Assigning Security Labels to Users

Just one security label (the default) can be stored in the user profile.

As security administrator, you should specify a default security label for each user that might need access to resources 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:

1. On z/VM, you can use the RACF ISPF panels to generate an EXEC using the SEARCH command with the CLIST operand.

How Users Specify Current Security Labels

On z/VM, when a user logs on and does not specify a security label on the CP LOGON command, the default security label stored in the user profile becomes the user's current security label. The user can

override the default security label by specifying the SECLABEL operand on the LOGON command as follows:

```
LOGON userid SECLABEL security-label
```

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 keyword 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 can use the RACSEC EXEC to display the current security label for your user ID or another user ID.

The syntax of the EXEC is as follows:

```
RACSEC {*|user ID}
```

*****
Use the asterisk (*) to specify that you want the current security label for your user ID.

user ID
Specify a user ID to obtain the current security label information for that user ID.

If you do not have a SECLABEL defined to your user ID, the following message is displayed:

```
RACSEC002I  Userid userid is not currently logged on, or does
            not have a security label.
```

If a SECLABEL is defined to your user ID, the following message is displayed:

```
RACSEC004I  The security label for user userid is seclabel.
```

To query the security label of a user other than yourself, you must meet one of the following requirements:

- If RACF protection for DIAGNOSE X'A0' subcode X'30' is in effect, then you must have READ access to the DIAG0A0.QUERYSEC profile in the VMCMD class.
- If RACF protection is not in effect for DIAGNOSE X'A0' subcode X'30', then you must have privilege class A or B.

For more information, see “Protecting the DIAGNOSE X'A0' Subcodes” on page 148.

Finding Out Which Security Labels a User Can Use

To find out which security labels a user can specify, issue the following command:

```
SEARCH  CLASS(SECLABEL)  USER(userid)
```

Note: Since SECLABELs must be RACLISTed, the class must be active before this SEARCH command can work.

Searching by Security Labels

To search for all profiles that have a particular security label, enter the following command:

```
SEARCH  CLASS(class-name)  SECLABEL(security-label)
```

For example:

```
SEARCH  CLASS(TERMINAL)  SECLABEL(EAGLE)
```

This command displays all the terminal profiles that have security label EAGLE specified.

```
SEARCH  CLASS(USER)      SECLABEL(EAGLE)
```

This command displays all the user profiles in which security label EAGLE is the default security label.

Note:

1. You can search only one class at a time.
2. RACF lists only the names of profiles to which the user has at least READ access authority.

SECLABEL Tranquility Considerations

For an evaluated Common Criteria 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.*

Preventing Changes to Security Labels

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.
- Prevent any user from changing the security label of a spool file with the CHANGE command.

To do this, enter the following command:

```
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 “Temporarily Preventing Significant RACF Activity” on page 259.

Temporarily Preventing Significant RACF Activity

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 the issuing of the RACINIT, RACHECK, or RACDEF macros, or an equivalent RACROUTE macro.)

To do this, enter the following command:

```
SETROPTS MLQUIET
```

To cancel this option, specify NOMLQUIET on the SETROPTS command.

SETROPTS Options Related to Security Labels

This section describes the SETROPTS options that require that the SECLABEL class be active.

Restricting Changes to Security Labels

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, issue the following command:

```
SETROPTS SECLABELCONTROL
```

When the SECLABELCONTROL option is in effect, only certain users can specify the SECLABEL operand on RACF commands:

- Users with the system-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 adding 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.

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 profile.

Preventing Users from Copying Data from One Security Label to a Lower Security Label

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 will still be able to copy data from a lower security label to the security label the user is currently logged on with.

To do this, enter the following command:

```
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.

Compatibility Mode for Security Labels

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 call was not made by a 1.9 or later caller, or the caller was not using or was unable to use the 1.9 or later 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 pre-1.9 call,” this means the ACEE (an area created to identify the call's security environment) was created by a pre-1.9 caller, or by a caller that was either not using or unable to use the 1.9 or later protocols. As a result, 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 1.9 or later protocols, you may be able to use the COMPATMODE keyword 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 the following command:

```
SETROPTS COMPATMODE
```

NOCOMPATMODE is in effect when a RACF database is first initialized using IRRMIN00.

Attention:
If you put COMPATMODE into effect, it affects all pre-1.9 callers.

Requiring Security Labels

If you have the SPECIAL attribute, and if the SECLABEL class is active, you can fully enforce multilevel security. 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 z/VM 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:

  DIRECTRY
  FILE
  TAPEVOL
  TERMINAL
  USER
  VMMDISK
  VMSEGMT
  WRITER

  To do this, enter the following command:

  ```
  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 forementioned classes and the resource has not been assigned a security label.

  To cancel the MLACTIVE option, specify NOMLACTIVE on the SETROPTS command.

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 system-SPECIAL attribute, specifying SYSHIGH as the current security label. Then either assign security labels, or issue SETROPTS NOMLACTIVE.

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 protect resources that need to be accessed when the user is logged on at different security labels (for example, a user's 191 minidisk).

Some products and applications depend on having a minidisk accessed in R/W mode to function properly. You should consider creating a minidisk and corresponding VMMDISK profile for each security label a user will be logging on with. By doing this, a user will always have at least one minidisk accessed in R/W mode when logged on.

For example, if SMITH needs to use security labels EAGLE and THRUSH, create VMMDISK profiles like:

```
SMITH.191   UACC(NONE)  SECLABEL(EAGLE)
SMITH.291   UACC(NONE)  SECLABEL(THRUSH)
```

Resources can be grouped into several categories and following certain procedures will minimize the impact of using them.

- **Read Only**

  In general, these resources pose no problem if they are protected by a profile with the lowest security label that a user will use. When 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**

  This type of resource should also be protected by a profile at the lowest security label that the user will use. 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, a file on a user's 291 minidisk) the user must log on at his or her lowest security label in order to update the file.

- **Read/Write**

  This type of resource should be protected by a profile at the security label that the user will most frequently use. When the SETROPTS MLS(FAILURES) option is in effect, access to these resources is prevented if the user is logged on with a security label different from the security label of the resource.

When RACF checks a user's authority to use a terminal, RACF uses "reverse MAC" (mandatory access checking). That is, the security label of the profile protecting the terminal must be equal to or greater than the security label of the user.

Activating the Security Label Class on z/VM

This section discusses the SECLABEL considerations that are specific for RACF on z/VM. For general information about SECLABELs with RACF, see, Chapter 17, “Security Classification and Zoning,” on page 249.

When the SECLABEL class becomes active, RACF indicates to z/VM that mandatory access control (MAC) has been enabled. z/VM then unconditionally calls RACF to determine MAC authorization for the subset of z/VM events which require MAC authorization.

When the SECLABEL class is active on a non-SSI system, z/VM does not call for MAC authorization *until* after one authorization or audit call is made to RACF. Once any call from z/VM to RACF has been made for

a z/VM event, z/VM begins calling RACF for MAC authorization. In other words, z/VM is not notified when the SECLABEL class is active. Instead, z/VM is notified when z/VM calls RACF next. For example, to get z/VM to call RACF at least once, you could do any of the following:

- Log on to any user (z/VM always calls RACF for logon authorization)
- Issue a LINK command (if you do not have a LINK/NOCTL member in an active VMXEVENT profile)
- Issue a command for which you have turned on VMXEVENT auditing.

While the SECLABEL class is active, control cannot be turned off selectively for the z/VM events that require MAC authorization. Unlike using VMXEVENT profiles and the SETEVENT command to control which z/VM events RACF is called to protect, when MAC authorization is in effect you cannot turn on z/VM calls for MAC authorization for some events and others off.

Table 42 on page 262 shows where to find discussions of how security label authorizations affect particular commands and classes.

Table 42. Where to Find Specific Security Label Information

Resource	Chapter Reference
LINK and MDISK Commands	“Security Label Checking for LINK and MDISK Events” on page 155
Unit Record Devices	“Security Label Considerations for Unit Record Devices” on page 157
RSCS Nodes	“Security Label Considerations for RSCS Nodes” on page 158
Alternate User IDs	“Security Label Considerations for Alternate User IDs” on page 160
VMSEGMT	“Security Label Considerations in the VMSEGMT Class” on page 162
Guest LAN	“Security Label Considerations for Guest LANs” on page 165
Terminals	“Using Security Labels to Control Terminals” on page 168
Profiles in the VMCMD Class	“Security Label Considerations for Profiles in the VMCMD Class” on page 143
SFS files and directories	“Security Label Considerations for SFS Files and Directories” on page 181
CP FOR command	“Protecting the FOR Command” on page 145

The following sections discuss security label considerations for other commands that are affected by MAC authorization.

LOGON, AUTOLOG, and XAUTOLOG Commands

z/VM always calls RACF for the LOGON, AUTOLOG, and XAUTOLOG commands. When the SECLABEL class is active and the users have a default SECLABEL defined in their user profiles, RACF ensures these users are authorized to the SECLABEL before allowing them to log on or be logged on.

When users are logged on using AUTOLOG or XAUTOLOG, they are assigned the default SECLABEL from their user profiles. When users log on with the LOGON command, a SECLABEL may be specified on the LOGON command; this SECLABEL is then used for authorization checking. If no SECLABEL is specified, the default SECLABEL for these users is used for authorization checking. For more information, see *z/VM: RACF Security Server General User's Guide*.

Additionally, when the SECLABEL and the VMMAC classes are active, an authorization check is made in the VMMAC class for the XAUTOLOG and AUTOLOG commands and when a user logs on to a logical device. See “Protecting z/VM Events with the VMMAC Class” on page 264 for more information.

User IDs Autologged During System Initialization

Several user IDs are automatically started by the z/VM operating system during system initialization. These user IDs include the following: AUTOLOG1, OPERACCT, OPEREREP, OPERSYMP, and OPERATOR. The autologged user IDs are tailorable using a system configuration file called SYSTEM CONFIG. RACF is not active until after these users have been logged on.

When these user IDs are logged on as part of z/VM initialization, z/VM assigns a SECLABEL of SYSHIGH to these users. If the SECLABEL class is active, for these user IDs to function properly once RACF is initialized, the system administrator should define SYSHIGH as the default SECLABEL for these users and permit the users to use the SYSHIGH security label.

For more information on SYSTEM CONFIG, see *z/VM Planning and Administration.*

Protecting a z/VM System Printer

When the SECLABEL class is active, z/VM calls RACF when a z/VM system printer is started. The printer operator (or issuer of the START command) specifies the security label of the data to be printed on the printer.

To start the printer, the security label of the data to be printed on the printer must be less than or equal to the security label of the printer. The security label of the printer is defined in the WRITER class profile for that printer.

In preparation for this authorization checking, you must do the following:

1. Define a profile in the WRITER class for each z/VM system printer, assigning a SECLABEL to the profile that represents the highest security level of data that can be printed on that printer. The printer name is the real device address of the printer, including leading zeros.

   For example, if the printer at device 0008 is allowed to print data up to and including the SECLABEL SECRET, define the profile as follows:

   ```
   RDEFINE  WRITER  0008  SECLABEL(SECRET)  UACC(NONE)
   ```

2. Authorize the appropriate users to be able to start the printer. READ access is required.

   ```
   PERMIT 0008 CLASS(WRITER) ID(USER01) ACCESS(READ)
   ```

3. Activate the WRITER class.

   ```
   SETROPTS CLASSACT(WRITER)
   ```

If your installation wants to use SECLABEL checking, but does not want to use SECLABEL checking for the WRITER class, deactivate the class by issuing:

```
SETROPTS NOCLASSACT(WRITER)
```

z/VM still calls for MAC authorization, but RACF defers the authorization to z/VM. The use of SECLABEL *NONE* is not allowed; it is assigned by z/VM when the printer is initialized. For restrictions on the use of a SECLABEL of NONE with z/VM system printers, see “Security Label NONE” on page 256.

Spool File Considerations

When the SECLABEL class is active, each spool file is assigned a security label when it is created. The security label assigned to the spool file is the security label at which the spool file creator is logged on. The security label is carried with the spool file instead of being stored in the RACF database. See also “Protecting z/VM Events with the VMMAC Class” on page 264 for more security label considerations for z/VM events that manipulate spool files.

Printing a Spool File

When the SECLABEL class is active, z/VM selects spool files to print based on the security label at which a printer is started (see "Protecting a z/VM System Printer" on page 263). For example, if the printer is started at the security label SECRET, only spool files with a security label of SECRET are allowed to print on that printer.

Once z/VM selects a file to be printed, z/VM calls RACF to see if the user is authorized to print the file. (The z/VM event for this call is UTLPRINT.) RACF examines the security label of the print spool file and determines if the user who printed the file is permitted to use the security label of the print spool file. This security label authorization is not related to the security label at which the user is currently logged on, because the user may be logged off when the file is actually printed.

Based on the security label authorization, if a user is not authorized to print a file, z/VM places the print file in SYSTEM HOLD status and sends a message to the system operator. The user is not notified of the print failure.

For example, if a user is authorized (through the PERMIT command) to use security labels SECRET and NOSECRET, but tries to print a spool file that is labeled SYSLOW, z/VM does not print the file. Instead, the file is placed in SYSTEM HOLD status and the system operator is notified.

Protecting z/VM Events with the VMMAC Class

When the SECLABEL class is active, the z/VM events requiring MAC authorization can be protected with the VMMAC class. To activate the VMMAC class, issue:

```
SETROPTS CLASSACT(VMMAC)
```

No profiles exist for this class. Instead, when authorization checking is performed for this class, RACF checks if the subject of the command (usually the command issuer) has the appropriate authority to the SECLABEL of the target resource. For details on the kind of SECLABEL authorization performed for each event, see Table 43 on page 264.

The outcome of each request depends on the SETROPTS options that are in effect. For more information, see "Security Label Authorization Checking" on page 313.

If your installation wants to use SECLABEL checking, but does not want to use SECLABEL checking for the VMMAC class, deactivate it by issuing:

```
SETROPTS NOCLASSACT(VMMAC)
```

z/VM still calls RACF for MAC authorization, but RACF defers the authorization to z/VM. z/VM treats this as a successful MAC authorization for these events.

Table 43. Security Label Authorizations in the VMMAC Class

z/VM Event	Privilege Class	RACF Class	Type of SECLABEL Author- ization	SECLABELs Compared
AUTOLOG	AB	USER and VMMAC	W/O	Issuer to autologee
CHANGE	G	VMMAC	W/O	Issuer to spool file
COUPLE	G	VMMAC	R/W	Issuer to target user (for CTCA)
FOR	C and G	VMMAC	R/W	Issuer to target user
LOGON (to LDEV)	ANY	USER and VMMAC	R/W	User logging on to creator of logical device
MESSAGE	ANY	VMMAC	W/O	Issuer to receiver
MSGNOH	B	VMMAC	W/O	Issuer to receiver
QUERY READER	G	VMMAC	R/O	Issuer to spool file
QUERY PRINTER	G	VMMAC	R/O	Issuer to spool file
QUERY PUNCH	G	VMMAC	R/O	Issuer to spool file

Table 43. Security Label Authorizations in the VMMAC Class (continued)

z/VM Event	Privilege Class	RACF Class	Type of SECLABEL Author- ization	SECLABELs Compared
QUERY TAG FILE	G	VMMAC	R/O	Issuer to spool file
QUERY TRFILES	ACDEG	VMMAC	R/O	Issuer to spool file
SMSG	G	VMMAC	W/O	Issuer to receiver
TAG FILE	G	VMMAC and VMNODE	W/O	Issuer to spool file
TAG QUERY FILE	G	VMMAC	R/O	Issuer to spool file
WARNING	ABC	VMMAC	W/O	Issuer to receiver
XAUTOLOG	AB	USER and VMMAC	W/O	Issuer to autologgee
XAUTOLOG	AB	USER and VMMAC	R/W	Autologgee to creator of logical device
XAUTOLOG	G	USER, VMMAC, and VMCMD	W/O	Issuer to autologee
DIAG014 (subcodes X'0004' X'0008' X' 0FFE' X'0FFF')	ANY	VMMAC	R/O	Issuer to spool file
DIAG068 (subcode X'000A' VMCF Identify)	ANY	VMMAC	R/W	Issuer to receiver
DIAG068 (subcode X'0002' VMCF Send)	ANY	VMMAC	W/O	Issuer to receiver
DIAG068 (subcode X'0003' VMCF Send and Receive)	ANY	VMMAC	R/W	Issuer to receiver
DIAG068 (subcode X'0004' VMCF Sendx)	ANY	VMMAC	W/O	Issuer to receiver
DIAG068 (subcode X'0005' VMCF Receive)	ANY	VMMAC	R/O	Issuer to receiver
DIAG068 (subcode X'0007' VMCF Reply)	ANY	VMMAC	W/O	Issuer to receiver
DIAG0B8 (subcode X'0000')	ANY	VMMAC	R/O	Issuer to spool file
DIAG0B8 (subcode X'0004')	ANY	VMMAC	W/O	Issuer to spool file
DIAG0BC	ANY	VMMAC	R/O	Issuer to spool file
DIAG23C (subcode X'03')	ANY	VMMAC	R/O or R/W	User being permitted to issuer
APPCCON	n/a	VMMAC	R/W	Issuer to target user
IUCVCON	n/a	VMMAC	R/W	Issuer to target user
SDF_OPEN[1]	n/a	VMMAC	R/O	Issuer to spool file
SPF_OPEN[1]	n/a	VMMAC	R/O	Issuer to spool file

Note:

1. An SDF_OPEN or SPF_OPEN can be issued by other z/VM events like DIAG0E0, DIAG034, or DIAG014. This can cause a SECLABEL authorization check for these other events.

Security Labels: An Example

Figure 23 on page 266 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 that they need read access to, but not write access)

In this example, there are two projects, A and B. Each project has one category (categories A and B). (While each project can have more than one category, it is simpler to show only one for each project.)

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.

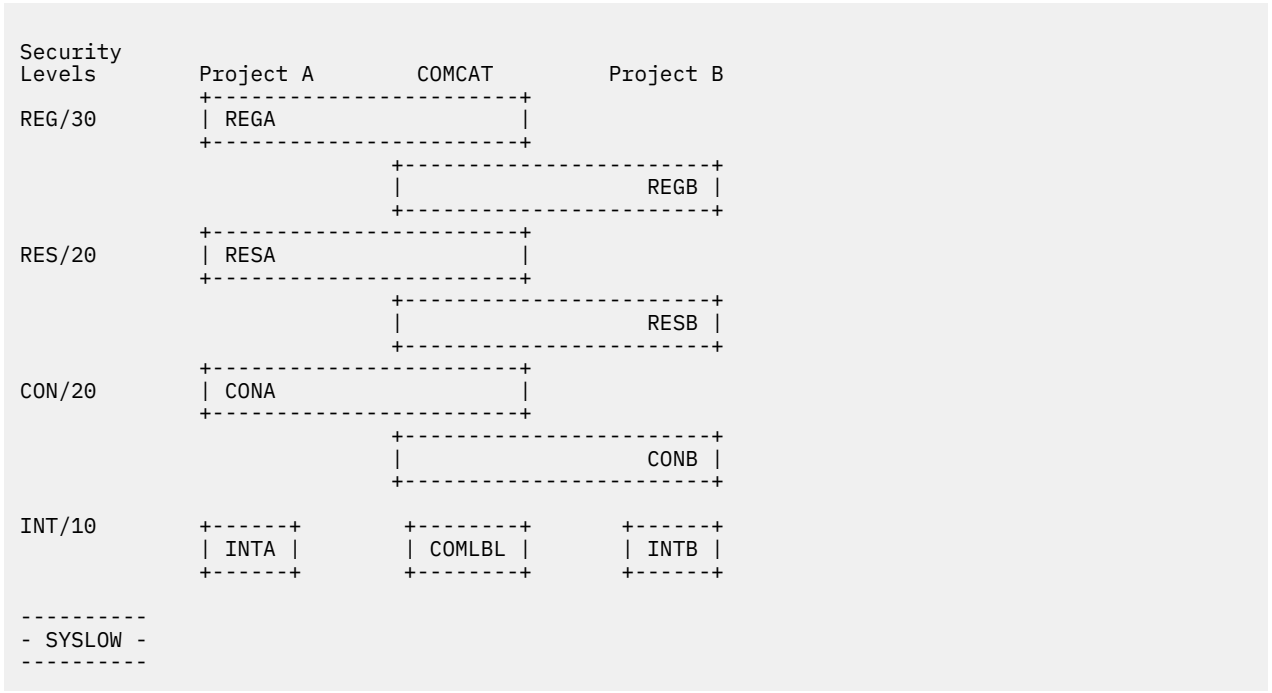


Figure 23. Security Labels: An Example

Note:

1. SYSHIGH includes security level REG and categories A, B, and COMCAT.
2. SYSLOW includes security level INT and no categories.

Note: 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 18. The RACF Database Unload Utility (IRRDBU00)

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 if the installation's security objectives are being met.

The RACF database unload utility enables installations to create a sequential file from a restructured 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 to process complex inquiries and create installation-tailored reports.

Diagnostic Capability

Although the design rational of the IRRDBU00 utility is not intended for diagnosis, this utility does provide some useful diagnostic capabilities. Since the IRRDBU00 utility must read every profile in the entire RACF database, it provides a side effect of validating profile data. While this is not a comprehensive validation of every field value, it is a validation of many lengths and count fields which are needed to successfully read each profile.

This side effect may 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 you do not receive this message, but rather abend or terminate in another fashion you may also be able to determine the profile in error. To do this, look in the output data set 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), is likely to be the culprit if indeed a bad profile is causing the utility to terminate.

See *z/VM: RACF Security Server System Programmer's Guide* and *z/VM: RACF Security Server Diagnosis Guide* for more information on RACF database diagnosis and correction.

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 restructured RACF database using the NOLOCKINPUT parameter.

As IRRDBU00 executes, it issues one RESERVE per profile, not a continuous RESERVE (this is also the case in IRRUT100 processing). When IRRDBU00 has finished copying a profile, it releases the RESERVE on it. 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.

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 profiles in the database. It does not unload all fields in each profile and treats some fields in a special way. Fields containing 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 FSROOT, HOME, and PROGRAM fields in the user's OVM segment are unloaded.

For the conversion rules of the database unload utility, see *z/VM: RACF Security Server Macros and Interfaces*.

The database unload utility uses the class descriptor tables (IBM-supplied and installation-defined) as it unloads profiles. If your database is imported from another system, you may also have to import the class descriptor tables (ICHRRCDX and ICHRRCDE). Classes will be 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/VM: RACF Security Server Macros and Interfaces*.

Using IRRDBU00 on z/VM (RACFDBU)

VM installations use the RACFDBU EXEC to execute the database unload utility.

You can execute the IRRDBU00 utility either by panel invocation or command invocation. For details, see "Panel Invocation of RACFDBU" on page 271 and "Command Invocation of RACFDBU" on page 272.

RACFDBU Setup

Before unloading your restructured RACF database, you must:

1. Log on to a virtual machine that has links to the RACF service machine's 490 disk and the RACF database disks (200, 300, and others).
2. Have at least READ access to the database to unload if using parameter NOLOCKINPUT.

   You must have WRITE access to the database to unload if using parameters UNLOCKINPUT or LOCKINPUT.

   For more information, see "Allowable Parameters" on page 274.
3. Ensure that the database to unload is restructured (block size is 4096).
4. IPL the RACF service machine's 490 disk.
5. Access the CMS 190 disk.
6. Ensure that there is adequate free space on the output minidisk to contain the utility output file.

   The size of the output file is roughly estimated as twice the size of the used portion of the input database, but you must also consider the type of profiles in your database. For example, profiles having variable length fields, such as installation data, require more space when unloaded because the maximum size of the field is unloaded (up to 255 bytes for most fields).

   Although the maximum length unloaded for most fields is 255 bytes, all 1023 bytes of data for the FSROOT, HOME, and PROGRAM fields in the user's OVM segment are unloaded.

   Determine the percentage of space your database is using by running the RACUT200 EXEC 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 will need approximately 70 free cylinders on your output minidisk.

   Note: On a z/VM system, there is no ability to span DASD packs; the size of minidisk output is restricted to one physical DASD device (minidisk).
7. Link the output minidisk as R/W.

Split Database Considerations

If your database has been split into several parts (maximum of 4), you may unload each part with separate utility invocations or you may unload all the parts with one utility invocation.

To unload all the parts of a split database with one utility invocation, you must enter the virtual addresses of *all* the database parts in your command or panel invocation. The order of the virtual addresses entered for the input databases must match the virtual addresses specified in the RACONFIG EXEC for the database parts.

To unload only one part of a split database with a utility invocation, you must enter the virtual addresses of the specific database part.

Panel Invocation of RACFDBU

Begin the exec by entering RACFDBU on the command line. The input panel appears on your screen. Figure 24 on page 271 illustrates the RACFDBU input panel.

```
        RACF Database Unload Utility - Input Panel


  . Status of input database:  1 = NOLOCKINPUT
                               2 = LOCKINPUT
                               3 = UNLOCKINPUT       i

  . Virtual address of input database                aaaa

  . Virtual address(es) of input database (optional)   bbbb cccc dddd

  . Virtual address of output minidisk               zzzz

  . Filename and filetype of output file          filename  filetype




               PF1 = Help    PF2 = Execute   PF3 = Quit
                   ENTER = Verify input fields
 Enter CP/CMS Commands below:

 ====>
```

Figure 24. Input Panel for RACFDBU

The user must supply the following values:

i
The status of the input database RACFDBU is processing.

For NOLOCKINPUT, enter the numeric 1.
For LOCKINPUT, enter the numeric 2.
For UNLOCKINPUT, enter the numeric 3.

See "Allowable Parameters" on page 274 for an explanation of these parameters. If UNLOCKINPUT is selected, it is not necessary to provide an output minidisk. UNLOCKINPUT unlocks the input database.

This field is required; 1 (NOLOCKINPUT) is the default.

aaaa
The virtual address of the input RACF database to unload. The blocksize of the input database must equal 4096. Using LOCKINPUT or UNLOCKINPUT on the database requires the user to have R/W access to this disk.

Note: If the input database is on FBA DASD that contains more than one file, the user will be prompted for the filename and filetype of the database to unload.

This is a required field.

bbbb cccc dddd
The virtual addresses of the input database to unload if the database has been split into parts. The block size of these input database parts must be 4096.

Note: FBA DASD does not support split databases, and if these fields are specified, they will be ignored.

These are optional input fields.

zzzz
The virtual address of a minidisk where the output from RACFDBU will be written.

This is a required field.

filename filetype
The file name and file type of the output file. RACFDBU OUTPUT is the default. You can supply another file name or file type.

If the output file you specify already exists, the utility changes the file type name of the existing file. For example, if the default file (RACFDBU OUTPUT) exists on the output minidisk, the existing file is copied to a file named RACFDBU OUTPUT1 on the same disk. It overlays any previous RACFDBU OUTPUT1 file. If the file type is 8 characters long, the last character is changed to a 1.

The input values entered on the panel are saved and reappear the next time you invoke RACFDBU. After you have entered your input in the required fields, press one of the following keys. The meaning of the ENTER key and the PF key definitions are:

Key
Meaning

Enter
Verify user input. Messages are issued if there are errors. (The database unload utility is not invoked.)

PF1
Display help screen.

PF2
Execute key. All input fields are validated and, if valid, the unload utility is invoked.

PF3
Exit panel.

Command Invocation of RACFDBU

Your installation may want to run the IRRDBU00 utility without interactive processing. To start the utility automatically, you can invoke the utility from a command line or a user-written exec, but input parameters must be correctly specified. The command invocation fields are similar to the panel invocation fields.

Note: If you are using FBA DASD and have more than one file on the minidisk, you must supply the file name and file type of the input RACF database.

All required parameters must be valid or the database unload utility will not be invoked.

The virtual addresses of an input RACF database and an output minidisk are required. There are defaults for ISTATUS, OUTFN, and OUTFT. If your database is on FBA DASD and there is more than one file on the minidisk, you must supply the file name and file type of the RACF database.

Syntax for command invocation

```
RACFDBU aaaa [bbbb [cccc [dddd ]]] zzzz [(options...[])]

Options:
[OUTFN filename]
[OUTFT filetype]
[ISTATUS i]
[FBAFN filename FBAFT filetype]
```

The explanation of the input fields follows:

aaaa
The virtual address of the input RACF database to unload. The block size of the input database must equal 4096. Using LOCKINPUT or UNLOCKINPUT on the database requires the user to have R/W access to this disk.

Note: If the input database is on FBA DASD that contains more than one file, the user must provide FBAFN and FBAFT.

This is a required field.

bbbb cccc dddd
The virtual addresses of the input database to unload if the database has been split into parts. The blocksize of these input database parts must be 4096.

Note: FBA DASD does not support split databases, and if these fields are specified, they will be ignored.

zzzz
Virtual address of output R/W minidisk.

This is a required field.

OUTFN *filename*
Filename of output CMS file.

This field is optional. RACFDBU is the default.

OUTFT *filetype*
Filetype of output CMS file.

This field is optional. OUTPUT is the default.

ISTATUS *i*
The status of the input database RACFDBU is processing.

For NOLOCKINPUT, enter the numeric 1.
For LOCKINPUT, enter the numeric 2.
For UNLOCKINPUT, enter the numeric 3.

See "Allowable Parameters" on page 274 for an explanation of these parameters.

This field is optional. 1 (NOLOCKINPUT) is the default.

FBAFN *filename*
Filename of RACF database on FBA DASD.

The field is required if there is more than one file on the input minidisk.

FBAFT *filetype*
Filetype of RACF database on FBA DASD.

The field is required if there is more than one file on the input minidisk.

Command Invocation Return Codes

To determine if the database unload utility successfully executed, check the return code. A return code of 0 indicates successful utility execution. A return code of 16 indicates that the utility did not execute. It is issued with error messages indicating the reason for failure.

IRRDBU00 Utility Messages on z/VM

Messages issued by the IRRDBU00 utility (IRR67xxx messages) are placed in a file named RACFDBU MESSAGES on the user's A-disk. The IRR67xxx messages are documented in *z/VM: RACF Security Server Messages and Codes*.

Messages from RACFDBU appear on the input screen and are documented in *z/VM: RACF Security Server Messages and Codes*. Messages issued by the RACFDBU EXEC begin with RPIDBU.

Allowable Parameters

One of the following parameters must be specified when running the database unload utility. On z/VM, indicate the parameter numerically: 1 for NOLOCKINPUT, 2 for LOCKINPUT, and 3 for UNLOCKINPUT.

For the *least* impact to system performance, use a copy of your restructured RACF database as input and specify the NOLOCKINPUT parameter.

Using the backup copy of the RACF database is allowed, as long as the backup is in the restructured format. Using an active copy of the restructured RACF database can impact system performance and it is not recommended.

- NOLOCKINPUT

  This 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 restructured RACF database as input and specify the NOLOCKINPUT parameter.

Attention:
If you use NOLOCKINPUT on the *active* database, your unloaded database may contain inconsistencies.

- LOCKINPUT

  This 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 database until the utility terminates.

  If you run IRRDBU00 and use LOCKINPUT, any activity updating the RACF database (such as users logging on and changing passwords or password phrases and batch jobs allocating new data sets requiring the creation of RACF profiles) will fail with either an ABEND483 RC50 or ABEND485 RC50.

  If using LOCKINPUT, do not schedule maintenance spanning midnight. If the RACF database remains locked past midnight, no new jobs will be able to be submitted, nor will users be able to 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.

  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 and IRRDSC00 utilities, which keep the input database locked and create new output databases. This is done to maintain integrity between the input database and the output database. IRRUT400 and IRRDSC00 assume that you no longer want or need the input database.

- UNLOCKINPUT

  UNLOCKINPUT is used to unlock a database that had been previously locked by the LOCKINPUT keyword. This action enables your input database and allows it to be updated.

  No data unloading is done when this parameter is used.

IRRDBU00 Output

The database unload utility reads every profile in a restructured RACF database. Depending on the contents of the profile, records of various types are created. All the records are written to a sequential file. For details on the format of the records created, see *z/VM: RACF Security Server Macros and Interfaces*.

This sequential output file can be:

- Used as input to a database management system, such as DB2.
- Manipulated with sort/merge utilities
- Used as input to your own programs
- Viewed directly

Sample SQL mappings are provided for z/VM installations, and the names of the files are:

- RACDBUTB SAMPLE
- RACDBULD SAMPLE
- RACDBUQR SAMPLE

See "Using the Database Unload Utility Output with SQL/DS" on page 275 for examples of using the IRRDBU00 output with SQL/DS.

Using the Database Unload Utility Output

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 will be able to produce reports tailored to their requirements.

Sort/Merge Programs

The database unload utility processes all the profiles in the input database. If you want a subset of the output records, you can use a standard utility such as DFSORT/CMS to select them.

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).

Using the Database Unload Utility Output with SQL/DS

The records produced by the database unload utility are designed to be processed by the SQL/DS Load Utility or its equivalent. The definition and control statements for a SQL/DS utilization of the output are as follows:

- Sample data definition language (DDL) statements to define the relational representation of the RACF database and sample SQL/DS definitions which perform database and index creation. These are in file RACDBUTB SAMPLE.
- Sample control statements for the SQL/DS Load Utility that map the output from the database unload utility (IRRDBU00). These are in file RACDBULD SAMPLE.
- Sample Structured Query Language (SQL) queries that perform useful data inquiries:
  - A query to list all the members of a group
  - A query to list all the users with the global-SPECIAL attribute
  - A query to find all the groups a user is connected to
  - A query to find all the data set access lists containing user IDs no longer valid.

  These are in file RACDBUQR SAMPLE.

  Note: The files RACDBUTB SAMPLE, RACDBULD SAMPLE, and RACDBUQR SAMPLE are shipped with RACF on the RACF service machine 305 disk.

For complete information on SQL/DS, see:

- *SQL/Data System General Information for IBM z/VM Systems, GH09-8074*
- *SQL/Data System Database Administration for IBM z/VM Systems, GH09-8083*
- *SQL/Data System System Administration for IBM z/VM Systems, GH09-8084*
- DB2 Server for VSE and VM SQL Reference (https://public.dhe.ibm.com/ps/products/db2vsevm/info/manuals75/db2v75r.pdf

Steps for Using IRRDBU00 Output with SQL/DS

To create and manage a SQL/DS database containing the database unload utility output, you must:

1. Create one or more SQL/DS DBSPACEs.
2. Create SQL/DS tables.
3. Create the SQL/DS indexes.
4. Load data into the tables.
5. Reorganize the indexes (optional).
6. 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 SQL/DS database, you erase your current table data. You then reload and reorganize your indexes.

The following sections show examples of the SQL/DS utility input for these functions.

Creating a SQL/DS DBSPACE

SQL/DS stores tables and indexes on tables in DBSPACEs. A DBSPACE is a logical allocation of space in the database. For more information see *SQL/DS System Administration*.

Creating the SQL/DS Tables

After the DBSPACE is created, SQL statements that define the tables are executed. Figure 25 on page 276 contains an example of the SQL statements required to create a table for the Group Basic Data record of the database unload utility.

The RACDBUTB SAMPLE file contains examples that create separate tables for each record type 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 GROUP_BD.
     ;
```

Figure 25. Sample SQL Utility Statements Creating a Table

Creating the SQL/DS Indexes

SQL/DS performance improves with the use of indexes. The RACDBUTB SAMPLE file creates an index for every primary key and every foreign key identified in the record types. Figure 26 on page 277 contains sample statements to create the indexes for the Group Basic Data record.

```
CREATE UNIQUE INDEX userid.GROUP_BD_IX1
       ON userid.GROUP_BD
          (GPBD_NAME)
          ;

CREATE        INDEX userid.GROUP_BD_IX2
       ON userid.GROUP_BD
          (GPBD_NAME, GPBD_SUPGRP_ID)
          ;

CREATE        INDEX userid.GROUP_BD_IX3
       ON userid.GROUP_BD
          (GPBD_OWNER_ID)
          ;

CREATE        INDEX userid.GROUP_BD_IX4
       ON userid.GROUP_BD
          (GPBD_MODEL)
          ;
```

Figure 26. Sample SQL Utility Statements Creating Indexes

Loading the SQL/DS Tables

Figure 27 on page 277 shows the statements required to load the Group Basic Data record. The RACDBULD SAMPLE file contains statements that load all the record types produced by the database unload utility. The sample requires that the output of RACFDBU be made into a fixed record length file.

```
DATALOAD TABLE (GROUP_BD)           IF POS(1:4)='0100'
          GPBD_NAME                  006-013
          GPBD_SUPGRP_ID             015-022
          GPBD_CREATE_DATE           024-033
          GPBD_OWNER_ID              035-042
          GPBD_UACC                  044-051
          GPBD_NOTERMUACC            053-053
          GPBD_INSTALL_DATA          058-311
          GPBD_MODEL                 314-357
          INFILE(IRRDBU00)
          )
```

Figure 27. SQL/DS Utility Statements Required to Load the Tables

Note: You can choose not to load some of the tables.

Reorganizing the Indexes in the SQL/DS Database

Queries are processed faster if they are performed against an organized database. SQL/DS provides a utility that allows you to reorganize the indexes on the catalog tables. For more information, see *SQL/DS Database Administration for z/VM*.

Deleting Data from the SQL/DS 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 DBSPACE statement for each DBSPACE.
3. Delete all the records in each table.

   Figure 28 on page 277 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 28. SQL Utility Statements Required to Delete the Group Records

SQL/DS Table Names

The RACDBUTB SAMPLE file creates SQL/DS tables for each record type. Table 44 on page 278 provides a useful reference of record type, record name, and SQL/DS table name.

Table 44. Correlation of Record Type, Record Name, and SQL/DS Table Name

Record Type	Record Name	SQL/DS 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
0130	Group OVM Data	GROUP_OVM_DATA
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
0210	User DFP Data	USER_DFP_DATA
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
02A0	User OVM Data	USER_OVM_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
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

Table 44. Correlation of Record Type, Record Name, and SQL/DS Table Name (continued)

Record Type	Record Name	SQL/DS Table Name
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

Samples Using the Database Unload Utility Output

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 data set profiles containing an invalid ID in the access list. This situation occurs when a security administrator deletes a user ID or group ID 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 IDs 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 IDs. A sample SQL query is provided in Figure 29 on page 280.
2. Format the results of the query as provided by the QMF form in Figure 30 on page 280.

The resulting report is shown in Figure 31 on page 281.

Note: If you used the IRRUT100 utility to check the references to a user or group ID, IRRUT100 requires that the user or group ID be known. The sample query does not have such a requirement. It finds *all* user IDs or group IDs that are not valid.

When your RACF database was unloaded, the IRRDBU00 utility created a Data Set Access Record (record type 404) for each user ID or group ID in the access list of each data set.

When you loaded your IRRDBU00 output into DB2, an AUTH_IDS table was created that contains the name of every valid user ID and group ID.

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 IDs (in AUTH_IDS). When a user ID is found that is not a valid user ID or group ID, 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 ID                                              --
--                                                                    --
-- Tables Accessed: SQL                                               --
--                  "DS_ACCESS"     - A list of dataset authorities   --
--                  "AUTH_IDS"      - A list of valid user/group IDs  --
--                                                                    --
-------------------------------------------------------------------------
          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 29. A Sample SQL Query

QMF Form

If the SQL query shown in Figure 29 on page 280 is processed using QMF, the data that is returned can be processed into a report. Figure 30 on page 280 shows a report or forms definition. It creates the report shown in Figure 31 on page 281 entitled "Data Set Profiles With Users Who Are Not Valid in the Access List".

```
 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 USERS WHO ARE NOT VALID IN THE
                        ACCESS LIST
           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 30. A Sample QMF Form

Report Output

This is the report that results from the SQL query shown in Figure 29 on page 280 and the QMF form shown in Figure 30 on page 280. Not all resulting rows are shown.

```
   DATA SET PROFILES WITH USERS WHO ARE NOT VALID IN THE ACCESS LIST

   DSACC_NAME                       DSACC_AUT DSACC_ACC  DSACC_ACCES
   -------------------------------  --------  ---------  -----------
   MARKN.WORK.CNTL                  WAYNEN    READ                 3
                                    DONE      READ                 2
                                    DANJ      READ                 2
                                    ANNEL     READ                 4
                                    TOMB      READ                 5
                                    ISABELH   READ                 2
                                    BOBS      READ                 1
                                    SUSANB    READ                 2
                                    DANL      READ                 3
                                    GLENS     READ                 3
                                    LOUL      READ                 6

   SUSANL.DOC.TEXT                  BOBS      READ                 4
                                    DANL      READ                 2

   TAMMY.TEST.DATA                  GLENS     READ                10

   WALT.DESIGN.DATA                 LOUL      READ                 3

   LAURIE.*                         DONE      UPDATE               6
                                    TOMB      READ                 1


 05/31/1995 04:26 PM                                        PAGE 10
```

Figure 31. A Sample Report

Chapter 19. 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 or password phrase called a PassTicket, which allows workstations and client machines to communicate with a host without using a RACF password or password phrase.

The secured signon function removes the need to send RACF passwords or password phrases 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/VM: RACF Security Server System Programmer's Guide*.

The RACF PassTicket

The RACF PassTicket is a *one-time-only*[1] 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)
```

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.

[1] 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.

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.

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 284).

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 285).

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

Depending on the application, the secured signon function uses a specific method for determining profile names in the PTKTDATA class. Information follows for using RACF secured signon for z/VM Control Program logons.

z/VM Control Program Logon

1. Ask the system programmer to determine the system ID by examining the CPU-ID (System ID) field in the RACF SMF CONTROL file.
2. Create the profile name to represent z/VM Control Program to the PTKTDATA class by prefacing the CPU-ID field with the characters VM.

Attention:
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.

Note: Because each application must be uniquely defined, you cannot use generic profiles to specify profiles to 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.

Protecting the Secured Signon Application Keys

When you define the secured signon application keys, RACF masks each key. See "Masking the Secured Signon Application Key" on page 285 for more information.

Note: Be sure the universal access authority (UACC) of the RACF data base is NONE. This prevents unauthorized users from listing or copying the RACF data set that contains the sensitive RACF secured signon application 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 application keys while they reside on the RACF data base is an IBM proprietary algorithm. It resides within the RACF object code portion of the RACF program product and 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 keys that the use of cryptography can provide.

To mask the application key when you define or alter it, use the SSIGNON operand and KEYMASKED suboperand with the RDEFINE or RALTER command.

Note: RACF for z/VM does not support the use of cryptographic products.

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/VM: RACF Security Server 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[2] indicating that the password is not valid.

[2] RACF sends a message to the appropriate log, 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 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:

   - A PassTicket is considered to be within the valid time range when the time on the generating computer's Time of Day (TOD) clock is within plus or minus 10 minutes of the time on the evaluating computer's TOD clock.

     Be sure that your systems which generate and evaluate PassTickets use TOD values that are within that time range.

     Synchronization of TODs can be accomplished using hardware-provided TOD synchronization features such as STP.

Attention:
Setting TOD clocks to UTC is not sufficient because TOD clocks can drift apart.

     For more information on setting clocks, see:

     – *z/VM CP Planning and Administration*
     – *z/VM Virtual Machine Operation*

     For more information on the RACF secured signon algorithms, see *z/VM: RACF Security Server Macros and Interfaces*.
   - If the value was used before, the user receives a message from the application[3] 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.
   - If the value is not valid, the host application sends a message[4] to the user indicating that the password is not valid.

Bypassing PassTicket Replay Protection

You might use the option to bypass PassTicket replay protection when the threat of PassTicket replay is not a security concern, such as in the following cases:

- Multiple end-users who share the same user ID
- Trusted registry domains that exchange PassTickets as a method of establishing trust
- Applications that request PassTickets for a particular USERID/APPLID combination more than once during a one-second time interval.

The option to bypass PassTicket replay protection 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.

[3] RACF sends a message to the appropriate log, 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.

[4] See the previous footnote.

The NO REPLAY PROTECTION text string will always be translated to upper case by the RALTER or RDEFINE commands.

The NO REPLAY PROTECTION text string can appear anywhere within the APPLDATA field, allowing for the existence of other information already in the field, or for new information that might be added in the future.

The following are examples of commands that will cause PassTicket replay protection to be bypassed.

Examples:

```
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')
```

Note:

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.

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” on page 287.

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 32. 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:

- 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 for more information.

Chapter 20. Authorizing Help Desk Functions

This topic describes how to authorize several common security tasks to the representatives of your installation's help desk, or customer call center.

Many installations delegate certain security tasks to help desk representatives in an effort to decentralize portions of user administration and reduce cost. The most commonly delegated tasks are meant to address the most frequently reported problems related to user security, such as forgotten passwords and logon failures.

In general, help desk representatives have less authority than security administrators. Security administrators are usually authorized with the SPECIAL or group-SPECIAL attribute, which gives them full authority over user profiles within their scope. By contrast, help desk representatives are not usually authorized with these attributes. Therefore, you must delegate to them the specific security tasks they need.

The following topics describe how to delegate the following authorities to the general users or groups on the staff of your installation's help desk or call center:

- "Delegating the Authority to List User Information" on page 289
- "Delegating the Authority to Reset Passwords and Password Phrases" on page 294.

Delegating the Authority to List User Information

You can authorize a general user or group to use the LISTUSER command to list information in the base segment of user profiles. You can choose to authorize a general user or group to list user information in *any* user profile (other than the profile of a user with the SPECIAL, OPERATIONS, AUDITOR, or ROAUDIT attribute), or you can limit the set of user profiles that a general user or group can list. In addition, you can delegate both authorities within your installation.

For details, see the following topics:

- "Delegating the Authority to List User Information in any User Profile" on page 289
- "Delegating the Authority to List User Information in only Selected User Profiles" on page 290

  When you limit the set of user profiles that a general user or group can list, you have the following options:

  - "Delegating the Authority to List User Information by Owner" on page 291
  - "Delegating the Authority to List User Information by Group Tree" on page 292
  - "Excluding Selected User Profiles" on page 293.

Delegating the Authority to List User Information in any User Profile

To authorize a general user to list user information in any user profile, define a profile to protect the IRR.LISTUSER resource in the FACILITY class. If you do not define this profile, standard LISTUSER authority checking applies when RACF determines whether the command issuer is authorized.

A general user can list the base segment of any user's profile when the command issuer has READ access to the IRR.LISTUSER resource in the FACILITY class. This authority authorizes a general user to list the base segment in the profile of any user including users with the PROTECTED attribute. **Restriction:** This authority does *not* apply when the target of the LISTUSER command has the SPECIAL, AUDITOR, ROAUDIT, or OPERATIONS attribute.

RACF does not log failed access attempts to IRR.LISTUSER. Successful accesses to IRR.LISTUSER are logged at the installation's discretion.

Steps for Delegating the Authority to List User Information in any User Profile

Before you begin: Make sure an existing generic profile in the FACILITY class does not inadvertently grant this authority.

Perform the following steps to authorize a general user or group to list user information in any user profile.

1. Define a profile to protect the IRR.LISTUSER resource in the FACILITY class.

   Example:

   ```
   RDEFINE FACILITY IRR.LISTUSER UACC(NONE)
      AUDIT(FAILURES(READ) SUCCESS(READ))
   ```

   _____

2. Authorize the general users or groups.

   Example:

   ```
   PERMIT IRR.LISTUSER CLASS(FACILITY) ID(HELPDESK USER19) ACCESS(READ)
   ```

   _____

3. Activate the FACILITY class if not already active.

   Example:

   ```
   SETROPTS CLASSACT(FACILITY)
   ```

   If the FACILITY class is already active and RACLISTed, refresh the FACILITY class profiles.

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

   _____

You have now authorized a general user or group to list the base segment of the user profile for any user, including a protected user, and excluding users with the SPECIAL, OPERATION, AUDITOR, or ROAUDIT attribute.

Delegating the Authority to List User Information in only Selected User Profiles

You can limit the authority of a general user or group to list user information by authorizing the user or group to list only a selected set of user profiles. You can limit the selected set of user profiles in the following ways:

- **Delegating by owner**

  You can limit the authority of a general user or group to list user information in user profiles based on the owner of the user profile. To do this, authorize the LISTUSER command issuer with READ authority to the IRR.LU.OWNER.*owner* resource in the FACILITY class.

  For details, see “Delegating the Authority to List User Information by Owner” on page 291.

- **Delegating by group tree**

  You can limit the authority of a general user or group to list user information in only user profiles that are within the scope of a selected group tree. To do this, authorize the LISTUSER command issuer with READ authority to the IRR.LU.TREE.*owner* resource in the FACILITY class.

  For details, see “Delegating the Authority to List User Information by Group Tree” on page 292.

- **Excluding user profiles**

  You can exclude selected user profiles from the scope of IRR.LU.OWNER.*owner* and IRR.LU.TREE.*owner* processing. To do this, protect the IRR.LU.EXCLUDE.*user-ID* resource in the FACILITY class.

  For details, see “Excluding Selected User Profiles” on page 293.

To authorize a general user or group to list user information in only selected user profiles, define a profile to protect the appropriate IRR.LU.OWNER or IRR.LU.TREE resource in the FACILITY class and grant READ access to authorize users and groups. If you do not define this profile, standard LISTUSER authority checking applies when RACF determines whether the command issuer is authorized.

The IRR.LU.OWNER and IRR.LU.TREE authorities authorize a general user to list the base segment in the profile of any user—based on owner or scope of the group tree—*including* protected users. **Restriction:** These authorities do not apply when the target of the LISTUSER command has the SPECIAL, AUDITOR, ROAUDIT, or OPERATIONS attribute.

RACF does not log failed access attempts to IRR.LU resources. Successful accesses to IRR.LU resources are logged at the installation's discretion.

Delegating the Authority to List User Information by Owner

You can authorize a general user or group to list user information in user profiles based on the owner of the user profile. To do this, authorize the LISTUSER command issuer with READ authority to the IRR.LU.OWNER.*owner* resource in the FACILITY class. The list-of-groups checking option (SETROPTS GRPLIST) need not be active and has no effect on this authority.

Steps for Delegating the Authority to List User Information by Owner

Before you begin:

- Make sure the LISTUSER command issuer does not have READ access to the IRR.LISTUSER resource in the FACILITY class.

Perform the following steps to limit the authority of a general user or group to list user information in selected user profiles based on the owner of the user profiles.

1. Define the following generic profiles in the FACILITY class, if not already defined. Doing so ensures that an existing generic profile does not inadvertently prevent you from successfully limiting this authority.

   Example:

   ```
   RDEFINE FACILITY IRR.LISTUSER.**   UACC(NONE)
   RDEFINE FACILITY IRR.LU.**         UACC(NONE)
   RDEFINE FACILITY IRR.LU.EXCLUDE.** UACC(READ)
   ```

2. Define a profile to protect the IRR.LU.OWNER.*owner* resource in the FACILITY class, where *owner* is the user ID or group that owns the user profiles.

   Example:

   ```
   RDEFINE FACILITY IRR.LU.OWNER.GROUP3 UACC(NONE)
      AUDIT(FAILURES(READ) SUCCESS(READ))
   ```

   _____

3. Authorize the general users or groups.

   Example:

   ```
   PERMIT IRR.LU.OWNER.GROUP3 CLASS(FACILITY) ID(HELPDESK USER19) ACCESS(READ)
   ```

   _____

4. Activate the FACILITY class if not already active.

   Example:

   ```
   SETROPTS CLASSACT(FACILITY)
   ```

   If the FACILITY class is already active and RACLISTed, refresh the FACILITY class profiles.

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

You have now authorized a general user or group to list the base segment of user profiles for selected users, including protected users, and excluding users with the SPECIAL, OPERATION, AUDITOR, or ROAUDIT attribute, based on the owner of the user profile.

Delegating the Authority to List User Information by Group Tree

You can authorize a general user or group to list user information in only user profiles that are within the scope of a selected group tree. To do this, authorize the LISTUSER command issuer with READ authority to the IRR.LU.TREE.*owner* resource in the FACILITY class, where *owner* is the group that is at the top of a group tree.

Rule: The list-of-groups checking option (SETROPTS GRPLIST) *must* be active.

Scope of a Group Tree

The scope of a group tree includes the following user profiles:

- User profiles that are owned by the group.
- User profiles that are owned by a subgroup that is owned by the group, or by a subgroup that is owned by a subgroup that is owned by the group, and so on.

The set of user profiles within scope of a group tree is the same set that applies when you authorize a user with the group-SPECIAL attribute. When you delegate by group tree, the user has authority to only view the base information in those user profiles. By contrast, when you give a user the group-SPECIAL attribute, the user has full authority over the user profiles within the scope of the group. For this reason, delegating by group tree is usually more appropriate for help desk personnel than authorizing them with the group-SPECIAL attribute.

Steps for Delegating the Authority to List User Information by Group Tree

Before you begin:

- Make sure the LISTUSER command issuer does not have READ access to the IRR.LISTUSER resource in the FACILITY class.
- Ensure that list-of-groups-checking (SETROPTS GRPLIST) is enabled.

Perform the following steps to authorize a general user to list user information in selected user profiles based on the scope of a group tree.

1. Define the following generic profiles in the FACILITY class, if not already defined. Doing so ensures that an existing generic profile does not inadvertently prevent you from successfully limiting this authority.

   Example:

   ```
   RDEFINE FACILITY IRR.LISTUSER.**   UACC(NONE)
   RDEFINE FACILITY IRR.LU.**         UACC(NONE)
   RDEFINE FACILITY IRR.LU.EXCLUDE.** UACC(READ)
   ```

2. Define a profile to protect the IRR.LU.TREE.*owner* resource in the FACILITY class, where *owner* is the group that is at the top of a group tree.

   Example:

   ```
   RDEFINE FACILITY IRR.LU.TREE.GROUP1 UACC(NONE)
      AUDIT(FAILURES(READ) SUCCESS(READ))
   ```

   ---

3. Authorize the general users or groups.

Example:

```
PERMIT IRR.LU.TREE.GROUP1 CLASS(FACILITY) ID(HELPDESK USER19) ACCESS(READ)
```

_____

4. Activate the FACILITY class if not already active.

   Example:

   ```
   SETROPTS CLASSACT(FACILITY)
   ```

   If the FACILITY class is already active and RACLISTed, refresh the FACILITY class profiles.

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

   _____

You have now authorized a general user or group to list the base segment of user profiles for selected users, including protected users, and excluding users with the SPECIAL, OPERATION, AUDITOR, or ROAUDIT attribute, based on the scope of a group tree.

Excluding Selected User Profiles

You can exclude selected user profiles from the scope of IRR.LU.OWNER.*owner* and IRR.LU.TREE.*owner* processing so that users authorized by these IRR.LU resources cannot list user information for the excluded user profiles. To exclude selected users, define a profile in the FACILITY class to protect the IRR.LU.EXCLUDE.*excluded-user* resource, where *excluded-user* is the user ID you are excluding.

When you protect the IRR.LU.EXCLUDE.*excluded-user* resource with UACC(NONE) and give no general users or groups access, the user information of the excluded user cannot be listed even when the command issuer has READ access to the appropriate IRR.LU.OWNER.*owner* and IRR.LU.TREE.*owner* resource in the FACILITY class.

In other words, when a general user, who has no access to the IRR.LU.EXCLUDE.*excluded-user* resource, attempts to list the user profile of an excluded user, the LISTUSER command fails.

Users and groups that you authorize with READ access to the IRR.LU.EXCLUDE.*excluded-user* resource are allowed to list the profile of the excluded user when they also have READ access to the appropriate IRR.LU resource.

Tip: If you want to exclude a set of users with similar user IDs, use a generic name (such as GRPADM*) in place of the excluded user ID.

Restriction: Users who are authorized by the IRR.LISTUSER resource are *not* limited when you exclude user profiles with the IRR.LU.EXCLUDE.*excluded-user* resource in the FACILITY class. Excluded users are excluded *only* when the general user or group has authority through the IRR.LU.OWNER.*owner* or IRR.LU.TREE.*owner* resource in the FACILITY class.

User profiles with the SPECIAL, AUDITOR, ROAUDIT, or OPERATIONS attribute cannot be listed by users with authority through the IRR.LU resources. Therefore, you need not exclude users with these attributes using the IRR.LU.EXCLUDE.*excluded-user* resource.

Steps for Excluding Selected User Profiles

Perform the following steps to exclude selected user profiles from the authority of a general user or group that is authorized through the IRR.LU.OWNER.*owner* or IRR.LU.TREE.*owner* resource in the FACILITY class.

1. Define the following generic profiles in the FACILITY class, if not already defined. Doing so ensures that an existing generic profile does not inadvertently prevent you from successfully excluding selected user profiles.

Example:

```
RDEFINE FACILITY IRR.LISTUSER.**   UACC(NONE)
RDEFINE FACILITY IRR.LU.**         UACC(NONE)
RDEFINE FACILITY IRR.LU.EXCLUDE.** UACC(READ)
```

2. Define a profile to protect the IRR.LU.EXCLUDE.*excluded-user* resource in the FACILITY class using UACC(NONE), where *excluded-user* is the user ID you want to exclude.

   Examples:

   ```
   RDEFINE FACILITY IRR.LU.EXCLUDE.SHANNON UACC(NONE)
      AUDIT(FAILURES(READ) SUCCESS(READ))
   RDEFINE FACILITY IRR.LU.EXCLUDE.GRPADM* UACC(NONE)
      AUDIT(FAILURES(READ) SUCCESS(READ))
   ```

   _____

3. Optionally, authorize selected users and groups with READ access to the IRR.LU.EXCLUDE.*excluded-user* resource. Perform this step only when certain users or groups who are authorized to an IRR.LU resource need to list the profile of the excluded user.

   Example:

   ```
   PERMIT IRR.LU.EXCLUDE.SHANNON CLASS(FACILITY) ID(HELPMGR) ACCESS(READ)
   ```

   _____

4. Activate the FACILITY class if not already active.

   Example:

   ```
   SETROPTS CLASSACT(FACILITY)
   ```

   If the FACILITY class is already active and RACLISTed, refresh the FACILITY class profiles.

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

   _____

You have now excluded selected user profiles from the authority of a general user or group that is authorized through the IRR.LU.OWNER.*owner* or IRR.LU.TREE.*owner* resource in the FACILITY class.

Delegating the Authority to Reset Passwords and Password Phrases

You can authorize a general user or group to use the ALTUSER command to resume user IDs and reset passwords and password phrases. You can choose to authorize a general user or group to do this for *any* user (other than users with the PROTECTED, SPECIAL, OPERATIONS, AUDITOR, or ROAUDIT attribute), or you can limit the set of users. In addition, you can provide both abilities within your installation.

For details, see the following topics:

- “Delegating the Authority to Reset the Password for any User” on page 296
- “Delegating the Authority to Reset Passwords for only Selected Users” on page 297

  When you limit the set of users, you have the following options:

  – “Delegating the Authority to Reset Passwords by Owner” on page 297
  – “Delegating the Authority to Reset Passwords by Group Tree” on page 298
  – “Excluding Selected Users” on page 300.

Levels of Authority

When you can delegate authority to a general user or group for resuming user IDs and resetting passwords and password phrases, define profiles in the FACILITY class to protect one or more of the following resources based on the scope of authority you need to delegate.

IRR.PASSWORD.RESET
Use this resource when the scope of authority includes all users.

IRR.PWRESET.OWNER.*owner*
Use this resource when the scope of authority is a limited set of selected users based on owner of the user ID.

IRR.PWRESET.TREE.*owner*
Use this resource when the scope of authority is a limited set of selected users based on scope of a group tree.

IRR.PWRESET.EXCLUDE.*excluded-user*
Use this resource to exclude a user profile from the scope of IRR.PWRESET.OWNER.*owner* and IRR.PWRESET.TREE.*owner* authority.

Restriction: You cannot delegate authority through the IRR.PASSWORD.RESET or IRR.PWRESET resources to authorize a general user or group to resume a revoked user or reset the password or password phrase for a user with *any* of the following attributes. Only users with the SPECIAL attribute, or the appropriate group-SPECIAL attribute, have resume and reset authorities for users with these attributes:

- SPECIAL
- OPERATIONS
- AUDITOR
- ROAUDIT
- PROTECTED.

Table 45. Authorities you can delegate based on the access level to the IRR.PASSWORD.RESET, IRR.PWRESET.OWNER, IRR.PWRESET.TREE, and IRR.PWRESET.EXCLUDE resources

Access authority to the IRR.PASSWORD.RESET IRR.PWRESET.OWNER IRR.PWRESET.TREE IRR.PWRESET.EXCLUDE resources	**Authorities for using the ALTUSER command that you can delegate to a general user or group**
READ	• Permits use of the PASSWORD operand to change a user's password to an expired password. **Restriction:** You cannot use the PASSWORD operand to add a password for a user who does not have one. • Permits use of the PHRASE operand to change a user's password phrase (and set as expired). **Restriction:** You cannot use the PHRASE operand to add a password phrase for a user who does not have one. • Permits use of the RESUME operand, without specifying a date, to resume a revoked user.
UPDATE	• Permits all authorities of READ access. • Permits use of the NOEXPIRED operand with the PASSWORD or PHRASE operand. (See **Notes**.)

Table 45. Authorities you can delegate based on the access level to the IRR.PASSWORD.RESET, IRR.PWRESET.OWNER, IRR.PWRESET.TREE, and IRR.PWRESET.EXCLUDE resources (continued)

Access authority to the IRR.PASSWORD.RESET IRR.PWRESET.OWNER IRR.PWRESET.TREE IRR.PWRESET.EXCLUDE resources	Authorities for using the ALTUSER command that you can delegate to a general user or group
CONTROL	• Permits all authorities of UPDATE access. • Permits use of the PASSWORD or PHRASE operand to reset a user's password or password phrase within the system's minimum change interval.

Note:

1. Neither being the owner of the user profile, nor having the group-SPECIAL attribute, provides sufficient authority to use the NOEXPIRED operand.
2. Only users who have the SPECIAL attribute can use the NOEXPIRED operand for users who have the SPECIAL, OPERATIONS, AUDITOR, or ROAUDIT attribute.

Delegating the Authority to Reset the Password for any User

To authorize a general user or group to use the ALTUSER command to resume a revoked user or reset a user's password or password phrase (other than for a protected user or a user with the SPECIAL, OPERATIONS, AUDITOR, or ROAUDIT attribute), define a profile to protect the IRR.PASSWORD.RESET resource in the FACILITY class. If you do not define this profile, standard ALTUSER authority checking applies when RACF determines whether the command issuer is authorized.

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.

Steps for Delegating the Authority to Reset the Password for any User

Perform the following steps to authorize a general user or group to use the ALTUSER command to resume a revoked user or reset a user's password or password phrase.

1. Define a profile to protect the IRR.PASSWORD.RESET resource in the FACILITY class.

   Example:

   ```
   RDEFINE FACILITY IRR.PASSWORD.RESET UACC(NONE)
      AUDIT(FAILURES(READ) SUCCESS(READ))
   ```

   _____

2. Authorize the general users or groups.

   Example:

   ```
   PERMIT IRR.PASSWORD.RESET CLASS(FACILITY) ID(HELPDESK USER19) ACCESS(READ)
   ```

   See “Levels of Authority” on page 295 for restrictions and details about authority based on the access level to IRR.PASSWORD.RESET.

   _____

3. Activate the FACILITY class if not already active.

Example:

```
SETROPTS CLASSACT(FACILITY)
```

If the FACILITY class is already active and RACLISTed, refresh the FACILITY class profiles.

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

———————————————————————————————————————

You have now authorized a general user or group to use the ALTUSER command to resume the user ID or reset the password or password phrase for any user, excluding protected users and users with the SPECIAL, OPERATION, AUDITOR, or ROAUDIT attribute.

Delegating the Authority to Reset Passwords for only Selected Users

You can limit the authority of a general user or group to use the ALTUSER command (to resume user IDs and reset passwords and password phrases) by authorizing the user or group to do this for only a selected set of users. You can limit the selected set of users in the following ways:

- **Delegating by owner**

  You can limit the authority of a general user or group to perform resume and reset functions based on the owner of the user profile. To do this, authorize the ALTUSER command issuer with the appropriate authority to the IRR.PWRESET.OWNER.*owner* resource in the FACILITY class.

  For details, see “Delegating the Authority to Reset Passwords by Owner” on page 297.

- **Delegating by group tree**

  You can limit the authority of a general user or group to perform resume and reset functions for only users within the scope of a selected group tree. To do this, authorize the ALTUSER command issuer with the appropriate authority to the IRR.PWRESET.TREE.*owner* resource in the FACILITY class.

  For details, see “Delegating the Authority to Reset Passwords by Group Tree” on page 298.

- **Excluding user profiles**

  You can exclude selected users from the scope of IRR.PWRESET.OWNER.*owner* and IRR.PWRESET.TREE.*owner* processing. To do this, protect the IRR.PWRESET.EXCLUDE.*user-ID* resource in the FACILITY class.

  For details, see “Excluding Selected Users” on page 300.

To authorize a general user or group to use the ALTUSER command to perform resume and reset functions for only selected users, define a profile to protect the appropriate IRR.PWRESET.OWNER or IRR.PWRESET.TREE resource in the FACILITY class and authorize users and groups. If you do not define this profile, standard ALTUSER authority checking applies when RACF determines whether the command issuer is authorized.

Restriction: The IRR.PWRESET.OWNER and IRR.PWRESET.TREE authorities do not apply when the target of the ALTUSER command is a protected user or has the SPECIAL, AUDITOR, ROAUDIT, or OPERATIONS attribute.

RACF does not log failed access attempts to IRR.PWRESET resources. Rather, these attempts are logged as ALTUSER command violations. Successful accesses to IRR.PWRESET resources are logged at the installation's discretion.

Delegating the Authority to Reset Passwords by Owner

You can authorize a general user or group to perform resume and reset functions for users based on the owner of the user profile. To do this, authorize the ALTUSER command issuer with the appropriate authority to the IRR.PWRESET.OWNER.*owner* resource in the FACILITY class. The list-of-groups checking option (SETROPTS GRPLIST) need not be active and has no effect on this authority.

Steps for Delegating the Authority to Reset Passwords by Owner

Before you begin:

- Make sure the ALTUSER command issuer does not have similar access to the IRR.PASSWORD.RESET resource in the FACILITY class.

Perform the following steps to limit the authority of a general user or group to resume user IDs and reset passwords and password phrases based on the owner of the user profiles.

1. Define the following generic profiles in the FACILITY class, if not already defined. Doing so ensures that an existing generic profile does not inadvertently prevent you from successfully limiting this authority.

   Example:

   ```
   RDEFINE FACILITY IRR.PASSWORD.RESET.**  UACC(NONE)
   RDEFINE FACILITY IRR.PWRESET.**         UACC(NONE)
   RDEFINE FACILITY IRR.PWRESET.EXCLUDE.** UACC(READ)
   ```

   If you use UPDATE or CONTROL access for any IRR.PWRESET profile, as described in Table 45 on page 295, specify the higher level (UPDATE or CONTROL) with the UACC operand for the `IRR.PWRESET.EXCLUDE.**` profile instead of the UACC(READ) option shown in this example.
2. Define a profile to protect the IRR.PWRESET.OWNER.*owner* resource in the FACILITY class, where *owner* is the user ID or group that owns the user profiles.

   Example:

   ```
   RDEFINE FACILITY IRR.PWRESET.OWNER.GROUP3 UACC(NONE)
      AUDIT(FAILURES(READ) SUCCESS(READ))
   ```

   ---

3. Authorize the general users or groups.

   Example:

   ```
   PERMIT IRR.PWRESET.OWNER.GROUP3 CLASS(FACILITY)
      ID(HELPDESK USER19) ACCESS(READ)
   ```

   See “Levels of Authority” on page 295 for restrictions and details about authority based on the access level to the IRR.PWRESET.OWNER.*owner* resource in the FACILITY class.

   ---

4. Activate the FACILITY class if not already active.

   Example:

   ```
   SETROPTS CLASSACT(FACILITY)
   ```

   If the FACILITY class is already active and RACLISTed, refresh the FACILITY class profiles.

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

You have now authorized a general user or group to resume user IDs and reset passwords and password phrases for selected users, excluding protected users, and users with the SPECIAL, OPERATION, AUDITOR, or ROAUDIT attribute, based on the owner of the user profile.

Delegating the Authority to Reset Passwords by Group Tree

You can authorize a general user or group to perform resume and reset functions for only users that are within the scope of a selected group tree. To do this, authorize the ALTUSER command issuer with the appropriate authority to the IRR.PWRESET.TREE.*owner* resource in the FACILITY class, where *owner* is the group that is at the top of a group tree.

Rule: The list-of-groups checking option (SETROPTS GRPLIST) *must* be active.

Scope of a Group Tree

The scope of a group tree includes the following user profiles:

- User profiles that are owned by the group.
- User profiles that are owned by a subgroup that is owned by the group, or by a subgroup that is owned by a subgroup that is owned by the group, and so on.

The set of user profiles within scope of a group tree is the same set that applies when you authorize a user with the group-SPECIAL attribute. When you delegate by group tree, the user has authority only to resume user IDs and reset passwords and password phrases. By contrast, when you give a user the group-SPECIAL attribute, the user has full authority over the users within the scope of the group. For this reason, delegating by group tree is usually more appropriate for help desk personnel than authorizing them with the group-SPECIAL attribute.

Steps for Delegating the Authority to Reset Passwords by Group Tree

Before you begin:

- Make sure the ALTUSER command issuer does not have similar access to the IRR.PASSWORD.RESET resource in the FACILITY class.
- Ensure that list-of-groups-checking (SETROPTS GRPLIST) is enabled.

Perform the following steps to limit the authority of a general user or group to resume user IDs and reset passwords and password phrases based on the scope of a group tree.

1. Define the following generic profiles in the FACILITY class, if not already defined. Doing so ensures that an existing generic profile does not inadvertently prevent you from successfully limiting this authority.

   Example:

   ```
   RDEFINE FACILITY IRR.PASSWORD.RESET.**  UACC(NONE)
   RDEFINE FACILITY IRR.PWRESET.**         UACC(NONE)
   RDEFINE FACILITY IRR.PWRESET.EXCLUDE.** UACC(READ)
   ```

   If you use UPDATE or CONTROL access for any IRR.PWRESET profile, as described in Table 45 on page 295, specify the higher level (UPDATE or CONTROL) with the UACC operand for the `IRR.PWRESET.EXCLUDE.**` profile instead of the UACC(READ) option shown in this example.
2. Define a profile to protect the IRR.PWRESET.TREE.*owner* resource in the FACILITY class, where *owner* is the group that is at the top of a group tree.

   Example:

   ```
   RDEFINE FACILITY IRR.PWRESET.TREE.GROUP1 UACC(NONE)
      AUDIT(FAILURES(READ) SUCCESS(READ))
   ```

   _____
3. Authorize the general users or groups.

   Example:

   ```
   PERMIT IRR.PWRESET.TREE.GROUP1 CLASS(FACILITY)
      ID(HELPDESK USER19) ACCESS(READ)
   ```

   See “Levels of Authority” on page 295 for restrictions and details about authority based on the access level to the IRR.PWRESET.TREE.*owner* resource in the FACILITY class.

   _____
4. Activate the FACILITY class if not already active.

Example:

```
SETROPTS CLASSACT(FACILITY)
```

If the FACILITY class is already active and RACLISTed, refresh the FACILITY class profiles.

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

You have now authorized a general user or group to resume user IDs and reset passwords and password phrases for selected users, excluding protected users, and users with the SPECIAL, OPERATION, AUDITOR, or ROAUDIT attribute, based on the scope of a group tree.

Excluding Selected Users

You can exclude selected user profiles from the scope of IRR.PWRESET.OWNER.*owner* and IRR.PWRESET.TREE.*owner* processing so that users authorized by these IRR.PWRESET resources cannot resume user IDs and reset passwords and password phrases for the excluded user profiles. To exclude selected users, define a profile in the FACILITY class to protect the IRR.PWRESET.EXCLUDE.*excluded-user* resource, where *excluded-user* is the user ID you are excluding.

When you protect the IRR.PWRESET.EXCLUDE.*excluded-user* resource with UACC(NONE) and give no general users or groups access, the excluded user's user ID cannot be resumed and the password and password phrase cannot be reset even when the command issuer has READ (or higher) access to the appropriate IRR.PWRESET.OWNER.*owner* and IRR.PWRESET.TREE.*owner* resource in the FACILITY class.

In other words, when a general user, who has no access to the IRR.PWRESET.EXCLUDE.*excluded-user* resource, attempts to resume the user ID or reset the password or password phrase of an excluded user, the ALTUSER command fails.

Users and groups that you authorize with READ access to the IRR.PWRESET.EXCLUDE.*excluded-user* resource are allowed to resume the user ID and reset the password and password phrase of the excluded user when they also have READ access to the appropriate IRR.PWRESET resource.

See “Levels of Authority” on page 295 for restrictions and details about authority based on the access level to the IRR.PWRESET.EXCLUDE.*excluded-user* resource in the FACILITY class.

Tip: If you want to exclude a set of users with similar user IDs, use a generic name (such as GRPADM*) in place of the excluded user ID.

Restriction: Users who are authorized by the IRR.PASSWORD.RESET resource are *not* limited when you exclude user profiles with the IRR.PWRESET.EXCLUDE.*excluded-user* resource. Excluded users are excluded *only* when the general user or group has authority through the IRR.PWRESET.OWNER.*owner* or IRR.PWRESET.TREE.*owner* resource.

Protected users and users with the SPECIAL, AUDITOR, ROAUDIT, or OPERATIONS attribute cannot be resumed, or have their passwords or password phrases reset, by users with authority through the IRR.PWRESET resources. Therefore, you need not exclude users with these attributes using the IRR.PWRESET.EXCLUDE.*excluded-user* resource.

Steps for Excluding Selected Users

Perform the following steps to exclude selected user profiles from the authority of a general user or group that is authorized through the IRR.PWRESET.OWNER.*owner* or IRR.PWRESET.TREE.*owner* resource in the FACILITY class.

1. Define the following generic profiles in the FACILITY class, if not already defined. Doing so ensures that an existing generic profile does not inadvertently prevent you from successfully excluding selected users.

Example:

```
RDEFINE FACILITY IRR.PASSWORD.RESET.**  UACC(NONE)
RDEFINE FACILITY IRR.PWRESET.**         UACC(NONE)
RDEFINE FACILITY IRR.PWRESET.EXCLUDE.** UACC(READ)
```

If you use UPDATE or CONTROL access for any IRR.PWRESET profile, as described in Table 45 on page 295, specify the higher level (UPDATE or CONTROL) with the UACC operand for the IRR.PWRESET.EXCLUDE.** profile instead of the UACC(READ) option shown in this example.

2. Define a profile to protect the IRR.PWRESET.EXCLUDE.*excluded-user* resource in the FACILITY class, where *excluded-user* is the user ID you want to exclude.

   Examples:

   ```
   RDEFINE FACILITY IRR.PWRESET.EXCLUDE.SHANNON UACC(NONE)
      AUDIT(FAILURES(READ) SUCCESS(READ))
   RDEFINE FACILITY IRR.PWRESET.EXCLUDE.GRPADM* UACC(NONE)
      AUDIT(FAILURES(READ) SUCCESS(READ))
   ```

   ---

3. Optionally, authorize selected users and groups with READ, UPDATE, or CONTROL access to the IRR.PWRESET.EXCLUDE.*excluded-user* resource, according to Table 45 on page 295. Perform this step only when certain users or groups who are authorized to an IRR.PWRESET resource need to resume the user ID or reset the password or password phrase of the excluded user.

   Examples:

   ```
   PERMIT IRR.PWRESET.EXCLUDE.SHANNON CLASS(FACILITY) ID(HELPMGR) ACCESS(READ)
   PERMIT IRR.PWRESET.EXCLUDE.GRPADM* CLASS(FACILITY) ID(HELPMGR) ACCESS(CONTROL)
   ```

   ---

4. Activate the FACILITY class if not already active.

   Example:

   ```
   SETROPTS CLASSACT(FACILITY)
   ```

   If the FACILITY class is already active and RACLISTed, refresh the FACILITY class profiles.

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

You have now excluded selected user profiles from the authority of a general user or group that is authorized through the IRR.PWRESET.OWNER.*owner* or IRR.PWRESET.TREE.*owner* resource.

Delegating Both by Owner and by Group Tree

You can delegate authority for the IRR.PWRESET and IRR.LU resources by profile owner, by group tree, or by a combination of both based on your installation's user profile structure.

If only a portion of your user profile structure is organized in a group-tree structure, use the IRR.LU.TREE and IRR.PWRESET.TREE resources to delegate help desk authorities to support users in that portion of the user population. For the portions of the user population that are not organized in a group-tree structure, use the IRR.LU.OWNER and IRR.PWRESET.OWNER resources to delegate help desk authorities.

Figure 33 on page 302 illustrates delegating help desk authorities by group tree (GROUP1) and delegating by owner (GROUP3) within the same RACF installation.

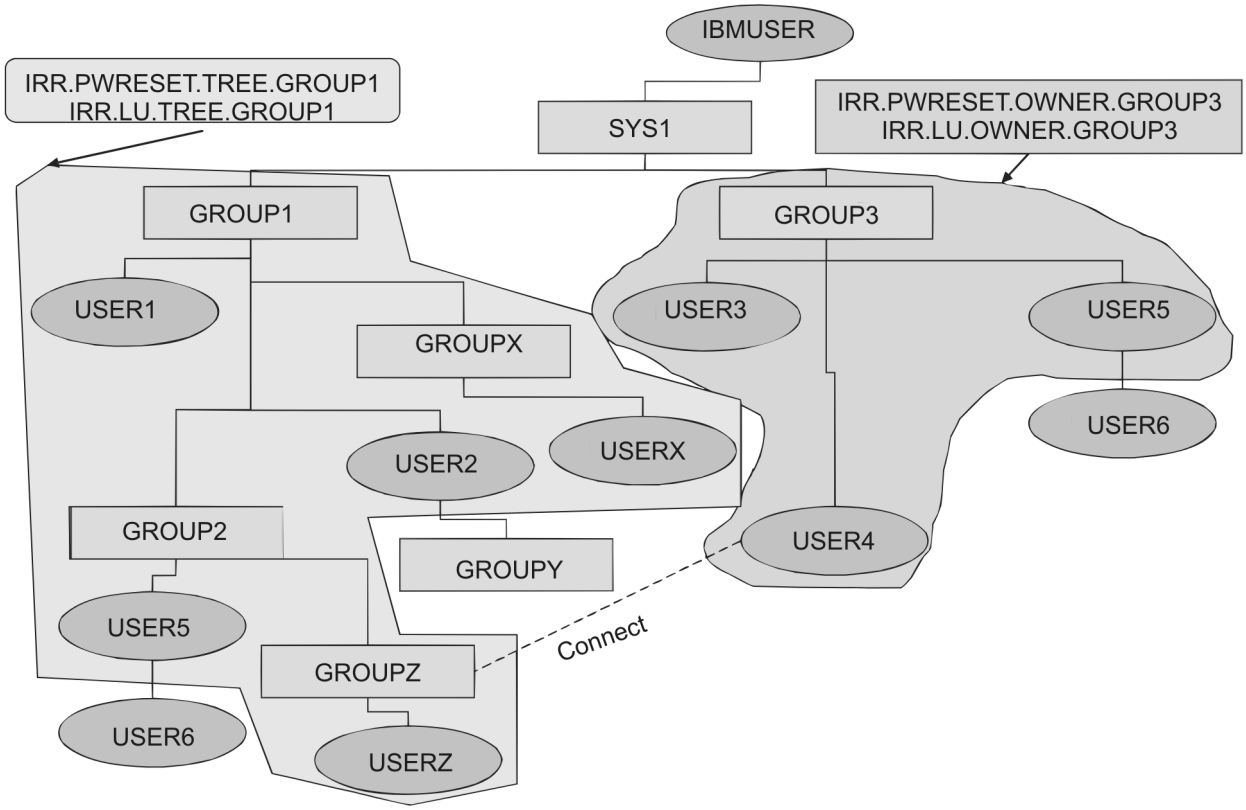


Figure 33. Sample group and user structure for delegating help desk authorities

Examples of Delegating Help Desk Authorities

This topic contains examples of delegating various help desk authorities.

Delegating Help Desk Authorities by Owner

The following examples delegate help desk authorities based on the owner of user profiles.

- User ANDREW needs the abilities to view user profile information, reset passwords and password phrases, resume user IDs, and use the NOEXPIRED operand for users that are owned by TEAMLDR.

  Examples:

```
RDEFINE FACILITY IRR.LU.OWNER.TEAMLDR UACC(NONE)
  AUDIT(FAILURES(READ) SUCCESS(READ))
PERMIT IRR.LU.OWNER.TEAMLDR CLASS(FACILITY) ACCESS(READ) ID(ANDREW)

RDEFINE FACILITY IRR.PWRESET.OWNER.TEAMLDR UACC(NONE)
   AUDIT(FAILURES(READ) SUCCESS(READ))
PERMIT IRR.PWRESET.OWNER.TEAMLDR CLASS(FACILITY) ACCESS(UPDATE) ID(ANDREW)

SETROPTS CLASSACT(FACILITY)
   or, if the FACILITY class is already active and RACLISTed:
   SETROPTS RACLIST(FACILITY) REFRESH
```

- The users connected to group HLPDESK8 need the abilities to view user profile information, reset passwords and password phrases, and resume user IDs for users that are owned by group AREA8. The following commands also prevent the user profile of the help desk administration user ID (HELPADM) from being listed and prevent its password from being reset.

  Examples:

```
RDEFINE FACILITY IRR.LU.OWNER.AREA8 UACC(NONE)
   AUDIT(FAILURES(READ) SUCCESS(READ))
PERMIT IRR.LU.OWNER.AREA8 CLASS(FACILITY) ACCESS(READ) ID(HLPDESK8)
RDEFINE FACILITY IRR.LU.EXCLUDE.HELPADM UACC(NONE)
```

```
RDEFINE FACILITY IRR.PWRESET.OWNER.AREA8 UACC(NONE)
   AUDIT(FAILURES(READ) SUCCESS(READ))
PERMIT IRR.PWRESET.OWNER.AREA8 CLASS(FACILITY) ACCESS(READ) ID(HLPDESK8)
RDEFINE FACILITY IRR.PWRESET.EXCLUDE.HELPADM UACC(NONE)

SETROPTS CLASSACT(FACILITY)
   or, if the FACILITY class is already active and RACLISTed:
   SETROPTS RACLIST(FACILITY) REFRESH
```

Delegating Help Desk Authorities by Group Tree

The following examples delegate help desk authorities based on the scope of a group tree.

- User USERH needs the abilities to reset passwords and password phrases and resume user IDs for users that are in the scope of group GROUP1.

  Examples:

```
RDEFINE FACILITY IRR.PWRESET.TREE.GROUP1 UACC(NONE)
   AUDIT(FAILURES(READ) SUCCESS(READ))
PERMIT IRR.PWRESET.TREE.GROUP1 CLASS(FACILITY) ACCESS(READ) ID(USERH)

SETROPTS CLASSACT(FACILITY)
   or, if the FACILITY class is already active and RACLISTed:
   SETROPTS RACLIST(FACILITY) REFRESH
```

- The users connected to group HLPDESK8 need the abilities to reset passwords and password phrases and resume user IDs for users that are in the scope of group GROUP1. The following commands also prevent the password of a group-SPECIAL user called USER1 from being reset.

  Examples:

```
RDEFINE FACILITY IRR.PWRESET.TREE.GROUP1 UACC(NONE)
   AUDIT(FAILURES(READ) SUCCESS(READ))
PERMIT IRR.PWRESET.TREE.GROUP1 CLASS(FACILITY) ACCESS(READ) ID(HLPDESK8)
RDEFINE FACILITY IRR.PWRESET.EXCLUDE.USER1 UACC(NONE)

SETROPTS CLASSACT(FACILITY)
   or, if the FACILITY class is already active and RACLISTed:
   SETROPTS RACLIST(FACILITY) REFRESH
```

Delegating Help Desk Authorities for All Users, Excluding Selected Users

In this scenario, an installation currently delegates the ability to reset passwords and list users to a group called HELPDESK by authorizing READ access to the IRR.PASSWORD.RESET profile and the IRR.LISTUSER profile in the FACILITY class. The installation wants to continue to delegate these abilities to the HELPDESK group but now wants to prevent the passwords of two users from being reset. In other words, users who are members of the HELPDESK group need to be authorized to reset passwords and list user profiles for all users except the group-SPECIAL users SHANNON and ANDREW.

The following examples remove the previous authorities from the HELPDESK group and then delegate the authority to reset passwords and list profiles for all users, excluding the two selected users.

1. Remove the HELPDESK group from the access list of the IRR.PASSWORD.RESET and IRR.LISTUSER profiles.

   Examples:

```
PERMIT IRR.PASSWORD.RESET CLASS(FACILITY) ID(HELPDESK) RESET
PERMIT IRR.LISTUSER CLASS(FACILITY) ID(HELPDESK) RESET

SETROPTS CLASSACT(FACILITY)
   or, if the FACILITY class is already active and RACLISTed:
   SETROPTS RACLIST(FACILITY) REFRESH
```

2. Delegate help desk authorities to the HELPDESK group using the IRR.LU and IRR.PWRESET profiles, excluding selected users.

Examples:

```
RDEFINE FACILITY IRR.LU.OWNER.* UACC(NONE)
   AUDIT(FAILURES(READ) SUCCESS(READ))
PERMIT IRR.LU.OWNER.* CLASS(FACILITY) ACCESS(READ) ID(HELPDESK)

RDEFINE FACILITY IRR.PWRESET.OWNER.* UACC(NONE)
   AUDIT(FAILURES(READ) SUCCESS(READ))
PERMIT IRR.PWRESET.OWNER.* CLASS(FACILITY) ACCESS(READ) ID(HELPDESK)

RDEFINE FACILITY IRR.PWRESET.EXCLUDE.ANDREW UACC(NONE)
RDEFINE FACILITY IRR.PWRESET.EXCLUDE.SHANNON UACC(NONE)

RDEFINE FACILITY IRR.LU.EXCLUDE.ANDREW UACC(NONE)
RDEFINE FACILITY IRR.LU.EXCLUDE.SHANNON UACC(NONE)

SETROPTS CLASSACT(FACILITY)
   or, if the FACILITY class is already active and RACLISTed:
   SETROPTS RACLIST(FACILITY) REFRESH
```

Note: In this scenario, there are no other profiles beginning with IRR.PWRESET.OWNER or IRR.LU.OWNER. If there are, then the HELPDESK group must be given READ access to each such profile.

Appendix A. When Problems Occur

The information in this appendix is provided for your use in debugging 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, not by your use of RACF, see *z/VM: RACF Security Server Diagnosis Guide* 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 will often have to 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. Turn to *z/VM: RACF Security Server Messages and Codes* for a description of the ICH408I message.

  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 isn't generic and would not be granting any access.
- If the ICH408I message indicates that access was denied because of a revoked user ID, you might want to resume that user ID.
- If the ICH408I message indicates that access was denied because the MFA server is unavailable, investigate whether there is a problem with the z/VM TCP/IP server configuration, the MFA server configuration, or the network connection. For more information, see “Problems with User ID Authentication” on page 317.
- 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) 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 SFS files, the user can issue the LFILE command.
  - For SFS directories, the user can issue the LDIRECT 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)
  - 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. This refreshes the user's copy of the profile.
- Issue the SETROPTS GENERIC(*class-name*) REFRESH or SETROPTS RACLIST(*class-name*) REFRESH command again.

SETROPTS REFRESH Processing on Shared Systems:
RACF does not automatically propagate the SETROPTS REFRESH command to other non-cluster systems sharing the database. This SETROPTS command must be issued to each system sharing the database. See "SETROPTS Command Propagation" on page 112. However, if you do not perform a refresh (issue the SETROPTS command with the REFRESH option) on a system sharing a RACF database and that system needs to re-IPL, the refresh takes effect on that system when re-IPL is performed.

- If access is being denied to a BFS file, use the `ls` command to display the file's permission bits and the file owner UID and GID. The user who has been denied access can issue the `id` command to display what the system is using as his or her UID, GID, and supplementary GID list. From this information, you can determine if RACF should allow access or not. If RACF is correctly denying access, you can change the file's permission bits using the `chmod` command to give the user access, if appropriate. If RACF should be allowing access based on the information shown, contact your IBM support center. The OpenExtensions shell commands `chmod`, `id`, and `ls` are documented in *OpenExtensions for z/VM: Command Reference.*
- 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 SMF data unload utility or 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 Unexpectedly

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.)

  Note: Even though RACF is active, resource managers for which external security is optional (such as SFS) 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 z/VM minidisks, check to see if there is an entry in the global minidisk table for the minidisk in question. An entry in the global minidisk table can allow access when a profile protecting the resource would deny access. Global disk table entries are defined in HCPRWA using the GLBLDSK macro. See *z/VM: RACF Security Server Macros and Interfaces* for details.
- If the resource is a general resource, make sure the general resource class is active. For example, for z/VM minidisks, make sure the VMMDISK 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 when a profile protecting the resource would deny access. For example, for minidisks, issue the following:

  ```
  RLIST GLOBAL VMMDISK
  ```

  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 that control is on for the z/VM event that is involved. For example, for accesses to other user's minidisks, make sure that control for LINK is on. To do this, issue the following command:

  ```
  SETEVENT LIST
  ```

  Also, if a USERSEL profile exists for the user, check to make sure that control for the command is turned on in the USERSEL profile. To do this, issue the following command:

  ```
  SETEVENT LIST USERSEL.userid
  ```
- 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 LDIRECT, LFILE, or RLIST command, specifying the resource name.
- 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 LDIRECT, LFILE, or RLIST command, specifying the resource name. 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 Resources Protected by RACF Profiles" on page 309 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 SFS files and directories, the second 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) 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. This refreshes the user's copy of the profile.
  - Issue the SETROPTS GENERIC(*class-name*) REFRESH or SETROPTS RACLIST(*class-name*) REFRESH command again.

SETROPTS REFRESH Processing on Shared Systems:
RACF does not automatically propagate the SETROPTS REFRESH command to other non-cluster systems sharing the database. This SETROPTS command must be issued to each system sharing the database. See "SETROPTS Command Propagation" on page 112. However, if you do not perform a refresh (issue the SETROPTS command with the REFRESH option) on a system sharing a RACF database and that system needs to re-IPL, the refresh takes effect on that system when re-IPL is performed.

- For resources defined in the VMCMD, VMMDISK, VMNODE, and VMRDR classes, the SYSSEC macro in HCPRWA can change the return code from the RACF service machine before giving control back to z/VM. Check your SYSSEC settings to determine if the access should be allowed. See *z/VM: RACF Security Server Macros and Interfaces* for details on the SYSSEC macro.
- You can use audit records to gather information that you wouldn't 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 SMF data unload utility or 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 Minidisk Profiles Take Effect

A change to a minidisk profile might not take effect immediately, particularly for users who have already linked to the minidisk. For example, if you remove a user from the access list, and that user has already linked to the minidisk, merely changing the user's access does not cause the minidisk to be detached from the user.

A change to a minidisk profile affects a user's access to the profile 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 CP QUERY command:

  ```
  QUERY userid
  ```

- If the user is logged on (or disconnected) and has not yet linked to the minidisk. You can check to see if a user is linked to a minidisk by linking to the minidisk yourself, and issuing the CP QUERY LINKS command:

  ```
  QUERY LINKS virtual-address
  ```

  Note: Linking to the minidisk yourself requires that you have access to the minidisk profile.

If the user is logged on (or disconnected) and has linked to the minidisk, and you change the user's access, two situations could occur:

- If the profile is a discrete profile, the user's access changes after detaching the minidisk.
- If the profile is a generic profile, the user's access changes after:
  - The user detaches the minidisk, if it is currently attached, and one of the following occurs when the user:
    - Issues SETROPTS GENERIC(VMMDISK) REFRESH (required if SETROPTS GENLIST(VMMDISK) is active)
    - Issues RL VMMDISK *userid.addr*
    - Enters a RACF command session
    - Logs off and then logs back on.

– The copy of the generic profile that is kept in virtual storage is changed.

The copy of the generic profile is changed when the user logs off and on again or when the SETROPTS GENERIC REFRESH command is issued.

Authorization Checking for Resources Protected by RACF Profiles

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 minidisk), the resource manager issues a RACROUTE REQUEST=AUTH request (or the RACHECK macro[5]). Based on the specifications on the RACROUTE REQUEST=AUTH request, or *RACF authorization request,* RACF determines whether the requesting user is authorized to access the resource.

- If the user is authorized to the resource, then 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, then 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), then 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 entry for the class.

  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.

Note:

1. SMF log records and/or messages may be generated, depending on the options in effect and whether RACF granted or denied access to the resource.
2. For spool data (protected by the VMRDR class), if the user ID of the requesting user is the user ID that created the spool file, RACF grants the request.
3. For z/VM events that are not controlled—on a system-wide basis or for particular users—RACF is not called for authorization.

Authorizing Access to Resources Protected by RACF Profiles

To perform authorization checking for RACF-protected resources, RACF makes the following checks. RACF stops processing when the request is granted or denied.

Note:

- Access to a minidisk may be granted by the global minidisk table prior to any of the following steps occurring. For more information on the global minidisk table, see *z/VM: RACF Security Server Macros and Interfaces*.

[5] If the RACHECK macro is issued instead of RACROUTE, authorization processing begins at step "5" on page 310.

- Access to an SFS file or directory may be granted by the SFSAUTOACCESS option prior to any of the following steps occurring. See *z/VM: RACF Security Server System Programmer's Guide* for more information.

  1. 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.
  2. If RACF is not active, RACF returns the "not protected" return code.
  3. For general resource classes, if the class of the resource is not active, RACF returns the "not protected" return code.
  4. If the RACF class must be RACLISTed (as specified in the class descriptor table) but is not currently RACLISTed, RACF returns the "not protected" return code.
  5. The RACROUTE REQUEST=AUTH preprocessing exit (ICHRCX01) can grant or deny the request.
  6. 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.
  7. If profiles for the class have not been brought into storage by RACLIST processing, and 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 RACROUTE REQUEST=AUTH request). If RACF finds a matching entry that allows access to the resource, RACF grants the request.
  8. 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 CDT (class descriptor table), described in *z/VM: RACF Security Server Macros and Interfaces*.) Specifically, no profile is found in the following cases:

     - Profiles for the class have been brought into common storage, but no profile in common storage matches the resource name.
     - If profiles for the class have *not* been brought into common storage, RACF looks for a profile in the user's storage. If no matching profile is found there, RACF looks 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), then RACF does not treat them as generic profiles. To recover, take the following steps:

     a. Issue SETROPTS NOGENERIC(*class-name*).
     b. Issue SETROPTS NOGENCMD(*class-name*).
     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(*class-name*).
     e. Define the profiles again.
  9. 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 313. If security label authorization checking succeeds, RACF authorization checking continues with step "11" on page 311.
  10. 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 "11" on page 311.

11. If users attempt to access their own resources, RACF grants the request.

    For SFS file and directories, if the user ID of the requesting user is the second qualifier of the file or directory name, RACF grants the request.

12. 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 "17" on page 311 (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).

13. 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 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 will use 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 "17" on page 311 (conditional access list checking). This prevents access based on ID(*), UACC, or the OPERATIONS attribute.

14. If a user ID of * is found on the standard access list, the current user is defined to RACF, and the access authority granted to * is sufficient to allow the requested access, RACF grants the request.
15. If the universal access authority (UACC) for the resource provides sufficient access authority for the requesting user to access the resource, RACF grants the request.
16. 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.
17. RACF checks the user's access authority in the conditional access list specified with WHEN(TERMINAL). 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.
18. 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). 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 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 will use 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.

19. If a user ID of * is found on the conditional access list specified with WHEN(TERMINAL), the current user is defined to RACF and 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.
20. If the WARNING flag is ON in the profile (set using the WARNING operand on the ADDDIR, ADDFILE, ALTDIR, ALTFILE, RALTER or RDEFINE command), RACF grants the request.
21. The RACROUTE REQUEST=AUTH postprocessing exit (ICHRCX02), if enabled, can grant or deny the request.
22. For resources defined in the VMCMD, VMMDISK, VMNODE, and VMRDR classes, the SYSSEC macro in HCPRWA can change the return code from the RACF service machine before giving control back to z/VM. See *z/VM: RACF Security Server Macros and Interfaces* for details on the SYSSEC macro.

Authorizing Access to RACF-Protected Terminals

When a RACF-defined user logs on to z/VM 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 RACINIT 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 313. If security label authorization checking succeeds, RACF authorization checking continues with the next step.
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 z/VM systems, see “Restricting Specific Groups of Users to Specific Terminals” on page 168.
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.)

Note:

1. The RACROUTE REQUEST=AUTH and RACROUTE 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 RACROUTE REQUEST=VERIFY.
3. Profiles in the GTERMINL class are ignored unless SETROPTS RACLIST processing is in effect.

Authorization Checking for RACROUTE REQUEST=FASTAUTH Requests

Some resource managers have high performance requirements. In order to do resource authorization checking with RACF, they make use of RACF facilities to load all the profiles for a given class into storage. Once these profiles are in storage, instead of doing a normal authorization check, the resource managers can do a "fast" authorization check.

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.
- 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.

Authorizing Access to RACF-Protected Applications

You can control access to some applications 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 Resources Protected by RACF Profiles” on page 309.
- If there is no matching profile in the APPL class, RACF allows the user to access the application.

Note:

1. The RACROUTE REQUEST=AUTH and RACROUTE 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 RACROUTE REQUEST=VERIFY.

Security Label Authorization Checking

Note: This sequence of authorization checks begins in one of the sequences in “Authorization Checking for Resources Protected by RACF Profiles” on page 309.

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.

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 requires security labels as defined in the class descriptor table, RACF fails the request.

Note: 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 RACROUTE 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.

   - **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* 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 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.

   - **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.)

   - **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 MESSAGE, SMSG, MSGNOH, and WARNING commands are write-only requests, as are some z/VMCF and IUCV requests.

     Note: VM systems do not support write-only requests for tape volumes or minidisks. 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 SQL tables, but these resources are not protected by RACF.

4. If the SETROPTS MLS(WARNING) option is in effect, RACF makes the same checks as in step "3" on page 314. If the access check fails on a read/write or write-only request because the user's current security label is greater than the 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 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 the categories (if any) used to define the security label of the resource are in the user's current security label.

     – All 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.

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.

Note: 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 46 on page 315 describes the results of security label authorization when the SECLABEL class is active and SETROPTS MLS(FAILURES) is active.

Table 46. 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 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:** The security label of the resource must be greater than the user's current security label. –or– The user's current security label must be greater than the security label of the resource. –or– 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 47 on page 316 describes the results of security label authorization when the SECLABEL class is active and SETROPTS NOMLS or SETROPTS MLS(WARNING) is active.

Table 47. 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 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) will be issued to the security console.

Authorization Summary for SETROPTS MLACTIVE

Table 48 on page 316 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 IBM-supplied class descriptor table defines which resource classes require security labels.)

Table 48. 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(WARNING) 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 will be passed/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 RACF 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 49 on page 317 shows the relationships of the security labels and the SETROPTS MLS, MLACTIVE(FAILURES) and MLQUIET options.

Table 49. 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 CDT, the DATASET class, and users must have security labels.
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” on page 317
- “Logon/Job Initialization Processing” on page 318.

When Logon or Job Initialization Processing Takes Place and Why

When a user requests access to a z/VM system, the application controlling the user's access can issue the RACROUTE REQUEST=VERIFY or REQUEST=VERIFYX macro.

On z/VM, some of the places RACROUTE REQUEST=VERIFY requests occur are:

- When users issue the CP LOGON command
- When users submit batch jobs through the VMBATCH facility.
- When service machines are autologged
- When a user enters a RACF command session.

Based on the specifications on the RACROUTE REQUEST=VERIFY request, RACF determines whether the requesting user is authorized to enter the system.

- If the user is authorized to enter the system, then 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, then RACF returns an "unauthorized" return code (other than 0) to the application. In general, the application then fails the request.

Note:

1. The RACROUTE REQUEST=VERIFY preprocessing and postprocessing exit routines are available during RACROUTE REQUEST=VERIFY 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). RACROUTE REQUEST=AUTH pre- and post-processing exits are available during this authorization processing.
3. SMF log records and/or messages can be generated (failures are always recorded; successes can be recorded if the application requests it on the RACROUTE 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 (such as ICH408I) that indicates the cause of the failure, such as:
  - User profile not defined
  - User ID revoked
  - Invalid or no password
  - Invalid group ID
  - Invalid or no security label (depending on RACF options)
  - Invalid or missing MFA CONTROL configuration
  - Invalid or missing configuration of the IBM Multi-Factor Authentication server

  If the application's message does not clearly indicate the source of the problem, review message ICH408I. This message, issued by RACF, might provide more information.

  If the user is enabled for MFA, RPIMFA messages, issued by RACF, might provide more information.

  Note:

  1. This message is either on the user's terminal or, for jobs, in the job log, or, if unavailable, it is on the security console or 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. 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.
- RACROUTE 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 50 on page 319 to see which resource classes could be checked from various types of sessions.

- Check if an installation exit is causing the problem. Candidates include:
  - Exits in the caller of RACF
  - The RACINIT exits

Table 50. Resource Classes Checked for Logon/Job Initialization Requests

Type of Session	Classes That Might Be Checked
VM logons	SECLABEL, SURROGAT, TERMINAL, VMMDISK, or other classes, depending on the user's z/VM logon procedure

Appendix B. Starting, Stopping, and Disabling RACF

This describes how to start, stop (temporarily suspend), reactivate, and disable RACF.

Starting and Restarting RACF

To start or restart a RACF service machine, for example RACFVM, you can use one of the following methods:

- Log on to RACFVM and enter:

```
#CP IPL 490
RACSTART
```

- From the primary system operator, enter:

```
FORCE RACFVM
XAUTOLOG RACFVM
```

- From a user defined as a secondary console of RACFVM, enter:

```
SEND CP RACFVM IPL 490
```

Note: It is not necessary to enter RACSTART because RACFVM is running disconnected; the RACSTART is performed automatically from within RACFVM's PROFILE EXEC.

Temporarily Suspending and Reactivating RACF

Use care when issuing the SETRACF command to deactivate RACF. When you deactivate RACF, access control reverts to CP. CP uses the information in the CP directory to control a user's access to the system (using the user's password) and to minidisks (using CP links). The information in the CP directory is probably not current with the equivalent information in the RACF database.

For example, if your installation changes a user's access authority to a minidisk from CONTROL to READ in the RACF data base, this change is not reflected automatically in the CP directory.

If you find it necessary to deactivate RACF, you should not allow general users to log on to the system while RACF is inactive.

SETRACF is a CMS command, not a RACF command; therefore you cannot enter SETRACF using RAC or during a RACF command session.

You can issue the SETRACF command only from a RACF service machine. The RACF service machine can, however run disconnected, thereby allowing a secondary console to issue this command.

By default, RACF sets up the OPERATOR as the secondary console for the RACF service machine; the OPERATOR can issue the command to deactivate RACF. For example,

```
SEND RACFVM SETRACF INACTIVE
```

If you issue SETRACF for any RACF service machine in a multiple service machine environment, it applies to all service machines.

For additional information about SETRACF, see *z/VM: RACF Security Server Command Language Reference*.

Temporarily Suspending RACF

To suspend RACF, enter:

```
SETRACF INACTIVE
```

The operator receives the following messages:

```
HCPRPD001I REQUEST TO SET RACF INACTIVE MADE BY RACFVM
HCPRPD007I RACF DEACTIVATION WILL FORCE SERVER IPL ON SUBSEQUENT ACTIVATION
HCPRPD002I REPLY YES TO ALLOW DEACTIVATION - ANYTHING ELSE WILL CANCEL REQUEST
```

Note: HCPRPD007I is returned only when running on an SSI.

The operator must respond YES.

Reactivating RACF

To reactivate RACF when it has been suspended, enter:

```
SETRACF ACTIVE
```

Disabling RACF On Your VM System

You can use the following procedure to disable RACF:

1. From OPERATOR user ID you need to FORCE RACFVM and XAUTOLOG RACMAINT. Need to do this so that PUT2PROD can run in the next step.
2. Regenerate the CP nucleus to replace the RACF modules with the VM modules: HCPRWA, HCPRPI, HCPRPF, HCPRPG, HCPRPD, HCPRPP, and HCPRPW. To do this the RACF local modification to CP needs to be removed. RACF also needs to be disabled. To accomplish this you need to issue the following commands from the MAINT*vrm* user ID:

   ```
   service racf disable
   put2prod
   ```
3. Remove XAUTOLOG statements for all RACF service machines from the AUTOLOG1 profile.
4. Move the functions from AUTOLOG2 back to AUTOLOG1.
5. Remove user LINKs to the RACFVM disks.
6. Modify DirMaint:
   a. Comment out the CONFIG DATADVH item ESM_PASSWORD_AUTHENTICATION_EXIT.
   b. Change the CONFIG DATADVH items PW_WARN_MODE and PW_LOCK_MODE to AUTOMATIC, if appropriate for your installation.
   c. Comment out the CONFIG DATADVH item ESM_LOG_RECORDING_EXIT.
   d. If using the sample RACF override CONFIGRC DATADVH file, RENAME it to CONFIGRC SAMPDVH. See the *z/VM Directory Maintenance Facility Tailoring and Administration Guide* for more information.

   For these changes to take effect, either IPL DIRMAINT or enter the DIRM RLDDATA command.
7. Remove RACF files from the 19E system disk, ISPF disk and GCS disk. You can use the VMFERASE command to remove those files if you used VMFCOPY to place the files on those disks. For example:

   ```
   ACC 19E Z
   VMFERASE PROD 7VMRAC30%RACF FROM Z
   ```
8. Modify SFS file pool servers.

   If you are using RACF to protect SFS resources, disable RACF support by changing the *server_id* DMSPARMS file on each SFS file pool server to specify NOESECURITY instead of ESECURITY.
9. Modify BFS file pool servers.

   If you are using RACF to protect BFS resources, disable RACF support by doing the following:

   a. Change the *server_id* DMSPARMS file on each BFS file pool server to specify NOESECURITY instead of ESECURITY.

b. Delete the file ESMLIB CSLLIB from each BFS file pool server. This file probably resides on the A-disk of the server.

c. In the PROFILE EXEC of each BFS file pool server, remove the `RTNLOAD DMSPERM` statement.

10. Modify RACROUTE applications.

    If you have applications or other program products which use the RACROUTE interface, you might have to change them. If you are replacing RACF with an equivalent security product, that product might or might not support the RACROUTE interface.

    To assist you in determining if any applications are using the RACROUTE interface, you can display which user IDs are authorized by RACF to issue RACROUTE requests. To do this, use the RLIST command to display the access list for the ICHCONN profile in the FACILITY class:

    ```
    RLIST FACILITY ICHCONN AUTH
    ```

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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 grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

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 jurisdictions 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 websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide 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 Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

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 document 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.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information may contain 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 actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information may contain sample application programs in source language, which illustrate 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. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., in the United States and/or other countries. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on IBM Copyright and trademark information (https://www.ibm.com/legal/copytrade).

The registered trademark Linux is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

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

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Terms and Conditions for Product Documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal Use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial Use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see:

- The section entitled **IBM Websites** at IBM Privacy Statement (https://www.ibm.com/privacy)
- Cookies and Similar Technologies (https://www.ibm.com/privacy#Cookies_and_Similar_Technologies)

Bibliography

This topic lists the publications in the z/VM library. For abstracts of the z/VM publications, see *z/VM: General Information*.

Where to Get z/VM Information

The current z/VM product documentation is available in IBM Documentation - z/VM (https://www.ibm.com/docs/en/zvm).

z/VM Base Library

Overview

- *z/VM: License Information*, GI13-4377
- *z/VM: General Information*, GC24-6286

Installation, Migration, and Service

- *z/VM: Installation Guide*, GC24-6292
- *z/VM: Migration Guide*, GC24-6294
- *z/VM: Service Guide*, GC24-6325
- *z/VM: VMSES/E Introduction and Reference*, GC24-6336

Planning and Administration

- *z/VM: CMS File Pool Planning, Administration, and Operation*, SC24-6261
- *z/VM: CMS Planning and Administration*, SC24-6264
- *z/VM: Connectivity*, SC24-6267
- *z/VM: CP Planning and Administration*, SC24-6271
- *z/VM: Getting Started with Linux on IBM Z*, SC24-6287
- *z/VM: Group Control System*, SC24-6289
- *z/VM: I/O Configuration*, SC24-6291
- *z/VM: Running Guest Operating Systems*, SC24-6321
- *z/VM: Saved Segments Planning and Administration*, SC24-6322
- *z/VM: Secure Configuration Guide*, SC24-6323

Customization and Tuning

- *z/VM: CP Exit Customization*, SC24-6269
- *z/VM: Performance*, SC24-6301

Operation and Use

- *z/VM: CMS Commands and Utilities Reference*, SC24-6260
- *z/VM: CMS Primer*, SC24-6265
- *z/VM: CMS User's Guide*, SC24-6266
- *z/VM: CP Commands and Utilities Reference*, SC24-6268

- *z/VM: System Operation*, SC24-6326
- *z/VM: Virtual Machine Operation*, SC24-6334
- *z/VM: XEDIT Commands and Macros Reference*, SC24-6337
- *z/VM: XEDIT User's Guide*, SC24-6338

Application Programming

- *z/VM: CMS Application Development Guide*, SC24-6256
- *z/VM: CMS Application Development Guide for Assembler*, SC24-6257
- *z/VM: CMS Application Multitasking*, SC24-6258
- *z/VM: CMS Callable Services Reference*, SC24-6259
- *z/VM: CMS Macros and Functions Reference*, SC24-6262
- *z/VM: CMS Pipelines User's Guide and Reference*, SC24-6252
- *z/VM: CP Programming Services*, SC24-6272
- *z/VM: CPI Communications User's Guide*, SC24-6273
- *z/VM: ESA/XC Principles of Operation*, SC24-6285
- *z/VM: Language Environment User's Guide*, SC24-6293
- *z/VM: OpenExtensions Advanced Application Programming Tools*, SC24-6295
- *z/VM: OpenExtensions Callable Services Reference*, SC24-6296
- *z/VM: OpenExtensions Commands Reference*, SC24-6297
- *z/VM: OpenExtensions POSIX Conformance Document*, GC24-6298
- *z/VM: OpenExtensions User's Guide*, SC24-6299
- *z/VM: Program Management Binder for CMS*, SC24-6304
- *z/VM: Reusable Server Kernel Programmer's Guide and Reference*, SC24-6313
- *z/VM: REXX/VM Reference*, SC24-6314
- *z/VM: REXX/VM User's Guide*, SC24-6315
- *z/VM: Systems Management Application Programming*, SC24-6327
- *z/VM: z/Architecture Extended Configuration (z/XC) Principles of Operation*, SC27-4940

Diagnosis

- *z/VM: CMS and REXX/VM Messages and Codes*, GC24-6255
- *z/VM: CP Messages and Codes*, GC24-6270
- *z/VM: Diagnosis Guide*, GC24-6280
- *z/VM: Dump Viewing Facility*, GC24-6284
- *z/VM: Other Components Messages and Codes*, GC24-6300
- *z/VM: VM Dump Tool*, GC24-6335

z/VM Facilities and Features

Data Facility Storage Management Subsystem for z/VM

- *z/VM: DFSMS/VM Customization*, SC24-6274
- *z/VM: DFSMS/VM Diagnosis Guide*, GC24-6275
- *z/VM: DFSMS/VM Messages and Codes*, GC24-6276
- *z/VM: DFSMS/VM Planning Guide*, SC24-6277

- *z/VM: DFSMS/VM Removable Media Services*, SC24-6278
- *z/VM: DFSMS/VM Storage Administration*, SC24-6279

Directory Maintenance Facility for z/VM

- *z/VM: Directory Maintenance Facility Commands Reference*, SC24-6281
- *z/VM: Directory Maintenance Facility Messages*, GC24-6282
- *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*, SC24-6283

Open Systems Adapter

- Open Systems Adapter/Support Facility on the Hardware Management Console (https://www.ibm.com/docs/en/SSLTBW_2.3.0/pdf/SC14-7580-02.pdf), SC14-7580
- Open Systems Adapter-Express ICC 3215 Support (https://www.ibm.com/docs/en/zos/2.3.0?topic=osa-icc-3215-support), SA23-2247
- Open Systems Adapter Integrated Console Controller User's Guide (https://www.ibm.com/docs/en/SSLTBW_2.3.0/pdf/SC27-9003-02.pdf), SC27-9003
- Open Systems Adapter-Express Customer's Guide and Reference (https://www.ibm.com/docs/en/SSLTBW_2.3.0/pdf/ioa2z1f0.pdf), SA22-7935

Performance Toolkit for z/VM

- *z/VM: Performance Toolkit Guide*, SC24-6302
- *z/VM: Performance Toolkit Reference*, SC24-6303

The following publications contain sections that provide information about z/VM Performance Data Pump, which is licensed with Performance Toolkit for z/VM.

- *z/VM: Performance*, SC24-6301. See z/VM Performance Data Pump.
- *z/VM: Other Components Messages and Codes*, GC24-6300. See Data Pump Messages.

RACF Security Server for z/VM

- *z/VM: RACF Security Server Auditor's Guide*, SC24-6305
- *z/VM: RACF Security Server Command Language Reference*, SC24-6306
- *z/VM: RACF Security Server Diagnosis Guide*, GC24-6307
- *z/VM: RACF Security Server General User's Guide*, SC24-6308
- *z/VM: RACF Security Server Macros and Interfaces*, SC24-6309
- *z/VM: RACF Security Server Messages and Codes*, GC24-6310
- *z/VM: RACF Security Server Security Administrator's Guide*, SC24-6311
- *z/VM: RACF Security Server System Programmer's Guide*, SC24-6312
- *z/VM: Security Server RACROUTE Macro Reference*, SC24-6324

Remote Spooling Communications Subsystem Networking for z/VM

- *z/VM: RSCS Networking Diagnosis*, GC24-6316
- *z/VM: RSCS Networking Exit Customization*, SC24-6317
- *z/VM: RSCS Networking Messages and Codes*, GC24-6318
- *z/VM: RSCS Networking Operation and Use*, SC24-6319
- *z/VM: RSCS Networking Planning and Configuration*, SC24-6320

TCP/IP for z/VM

- *z/VM: TCP/IP Diagnosis Guide*, GC24-6328
- *z/VM: TCP/IP LDAP Administration Guide*, SC24-6329
- *z/VM: TCP/IP Messages and Codes*, GC24-6330
- *z/VM: TCP/IP Planning and Customization*, SC24-6331
- *z/VM: TCP/IP Programmer's Reference*, SC24-6332
- *z/VM: TCP/IP User's Guide*, SC24-6333

Prerequisite Products

Device Support Facilities

- Device Support Facilities (ICKDSF): User's Guide and Reference (https://www.ibm.com/docs/en/SSLTBW_2.5.0/pdf/ickug00_v2r5.pdf), GC35-0033

Related Products

XL C++ for z/VM

- *XL C/C++ for z/VM: Runtime Library Reference*, SC09-7624
- *XL C/C++ for z/VM: User's Guide*, SC09-7625

z/OS

IBM Documentation - z/OS (https://www.ibm.com/docs/en/zos)

Index

Special Characters

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

Product Number: 5741-A09

Printed in USA

SC24-6311-74