DB2 Universal Database™ for z/OS

Version 8

RACF Access Control Module Guide

Information Management software

SC18-7433-05

DB2 Universal Database™ for z/OS

Version 8

RACF Access Control Module Guide

Information Management software

SC18-7433-05

Note

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 107.

Sixth Edition, Softcopy Only (October, 2009)

This edition applies to Version 8 of IBM Universal Database for z/OS (DB2 UDB for z/OS), 5625-DB2, and to any subsequent releases until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

This softcopy version is based on the printed edition of the book and includes the changes indicated in the printed version by vertical bars. Additional changes made to this softcopy version of the book since the hardcopy book was published are indicated by the hash (#) symbol in the left-hand margin. Editorial changes that have no technical significance are not noted.

This and other books in the DB2 UDB for z/OS library are periodically updated with technical changes. These updates are made available to licensees of the product on CD-ROM and on the Web (currently at http://www.ibm.com/software/data/db2/zos/library.html). Check these resources to ensure that you are using the most current information.

Contents

About this document

This document contains information about planning, installing, and implementing the RACF® access control module, a sample exit routine called DSNXRXAC, that ships with DB2 Universal Database for z/OS Version 8. You can replace the default DB2® exit routine at the access control authorization exit point (DSNX@XAC) with the RACF access control module to use RACF, in addition to DB2, for authorization checking of access to DB2 objects, authorities, commands, and utilities.

Who should use this document

Use this document as a guide to the task of planning, installing, and implementing the RACF access control module. The skills required include MVS™ system programming, DB2 administration, and RACF administration. The participants for this task should include those who are knowledgeable in the current security structure and policies in place for both DB2 and RACF at your installation.

Where to find more information

The following resources are available through the Internet.

- Online library for DB2 Universal Database for z/OS® Version 8:

  http://www.ibm.com/software/data/db2/zos/v8books.html
- DB2 home page:

  http://www.ibm.com/software/data/db2/zos/
- Online library for z/OS Security Server RACF:

  http://www.ibm.com/servers/eserver/zseries/zos/bkserv/
- RACF home page:

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

Summary of changes to this document

This section summarizes the major changes to this document for Version 8.

The following chapters have changed to support the RACF dynamic class descriptor table (dynamic CDT) introduced with z/OS Version 1 Release 6.
- Chapter 1, “Overview,” on page 1
- Chapter 2, “Planning,” on page 5
- Chapter 4, “Defining classes for the RACF access control module (optional),” on page 17.

“Migrating from the IRR@XACS module” on page 7 is added.

Chapter 3, “Installing the RACF access control module” has changed as follows: Step 1 of “Steps for testing that your exit routine is active” on page 14 is corrected.

Table 5 on page 25 documents an additional object name qualifier for view objects.

Chapter 7, “Making your new RACF resources effective,” on page 29 is corrected.

“Authorization checking for operations on views” on page 43 is added.

Appendix D, “RACF authorization checking reference,” on page 67 has changed as follows:
- Authorization check documentation for the following privileges is corrected:
  - ALTERIN, COMMENT ON, and DROPIN privileges for schemas
  - ALTER and COMMENT ON privileges for sequences
  - DISPLAY and EXECUTE privileges for stored procedures
  - DISPLAY and EXECUTE privileges for user-defined functions.
- Authorization checking for the DELETE, INSERT, and UPDATE privileges for views has changed.
- “How to set the level of access” on page 70 is added.

Chapter 1. Overview

The RACF access control module allows you to use RACF in addition to DB2 authorization checking for DB2 objects, authorities, commands, and utilities. You can activate the RACF access control module at the DB2 access control authorization exit point (DSNX@XAC), where you can replace the default DB2 exit routine.

The RACF access control module requires DB2 Version 8, or later, and z/OS Version 1 Release 3, or later, and is supplied as an assembler source module in the DSNXRXAC member of *prefix*.SDSNSAMP. Support for earlier DB2 versions was supplied in the IRR@XACS member of SYS1.SAMPLIB and was called the *DB2 RACF external security module.*

The RACF access control module:
- Receives control from the DB2 access control authorization exit point (DSNX@XAC) to handle DB2 authorization checks
- Provides a single point of control for RACF and DB2 security administration
- Provides the ability to define security rules before a DB2 object is created
- Allows security rules to persist when a DB2 object is dropped
- Provides the ability to protect multiple DB2 objects with a single security rule using a combination of RACF generic, grouping, and member profiles
- Eliminates the DB2 *cascading revoke*
- Preserves DB2 privileges and administrative authorities
- Provides flexibility for multiple DB2 subsystems with a single set of RACF profiles
- Allows you to validate a user ID before giving it access to a DB2 object.

RACF support for the RACF access control module includes:
- A set of general resource classes in the RACF module ICHRRCDX (the supplied portion of the RACF class descriptor table). These classes are used when you implement the RACF access control module using the default values.
- # With z/OS Version 1 Release 5 and earlier, a set of router table entries in the
  # RACF module ICHRFR0X (the supplied portion of the SAF router table).

  # **Note:** Beginning with z/OS Version 1 Release 6, router table entries are no
  # longer required, and the RACF module ICHRFR0X is no longer shipped.

RACF checking for DB2 resources

Each DB2 command, utility, and Structure Query Language (SQL) statement is associated with a set of privileges, authorities, or both.

Authority checking performed by the RACF access control module simulates DB2 authority checking:
- DB2 object types *map to* RACF class names
- DB2 privileges *map to* RACF resource names for DB2 objects
- DB2 authorities *map to* the RACF administrative authority class (DSNADM) and RACF resource names for DB2 authorities
- DB2 security rules *map to* RACF profiles

The RACF access control module checks the RACF profiles corresponding to that set of privileges and authorities.

See Chapter 10, “Special considerations,” on page 41 and Appendix D, “RACF authorization checking reference,” on page 67 for more information.

Multilevel security

Multilevel security is a security policy that allows the classification of data and users based on a system of hierarchical security levels combined with a system of non-hierarchical security categories. You can improve the security of your DB2 applications when you add RACF security labels to DB2 objects or row-level security on a multilevel-secure system. Implementing multilevel security requires the use of z/OS Version 1 Release 5, or later, and is a system-wide endeavor. See *z/OS Planning for Multilevel Security and the Common Criteria*, GA22-7509 for details.

This document does not address the use of DB2 and the RACF access control module in a multilevel-secure environment.

The DB2 access control authorization exit point

DB2 provides an exit point so you can install the RACF access control module to allow RACF to perform DB2 authorization checking for SQL statements and DB2 commands and utilities, or you can choose to provide your own routine for the DB2 access control authorization exit point. This document describes how to implement only the supplied RACF access control module. For more information about providing your own routine for the DB2 access control authorization exit point, see *DB2 Administration Guide*.

The default DB2 exit routine

The default DB2 exit routine at the DSNX@XAC exit point returns a code to the DB2 authorization module indicating that an installation-defined access control authorization exit routine is not available. DB2 then performs native authorization checking and does not attempt to invoke this exit routine again. The default DB2 exit routine called DSNX@XAC is in library *prefix*.SDSNLOAD. The source code for the default DB2 exit routine is in the DSNXSXAC member of *prefix*.SDSNSAMP. The DB2 installation process puts the results of the assembly into *prefix*.SDSNEXIT.

By contrast, the RACF access control module is provided in DSNXRXAC member of *prefix*.SDSNSAMP and provides access control using a combination of RACF and DB2 checking. You can easily alter the DB2 installation process by modifying the DSNTIJEX job to assemble the RACF access control module, rather than the default DB2 exit routine.

When the RACF access control module is invoked

The RACF access control module is invoked in three instances:

- At DB2 startup

  When DB2 starts, the RACF access control module is invoked to allow the external authorization checking application to perform any required setup prior to authorization checking. An example of a required setup task is loading authorization profiles into storage. DB2 uses the reason code that the exit routine sets during startup to determine how to handle exception situations. (See *DB2 Administration Guide* for details.)

- When an authorization check is to be performed for a privilege

  At the point when DB2 would access security tables in the catalog, to check authorization on a privilege, the RACF access control module is invoked. The exit routine is only invoked if none of the prior invocations have indicated that the exit routine must not be called again.
- At DB2 shutdown

  When DB2 is stopping, the RACF access control module is invoked to let the external authorization checking application perform its cleanup before DB2 stops.

When the RACF access control module is bypassed

In the following situations, the RACF access control module is not called to check authorization:

- The user has installation SYSOPR (when sufficient for the privilege being checked) or installation SYSADM authority. This authorization check is made strictly within DB2.
- DB2 security has been disabled (`NO` was specified in the USE PROTECTION field of installation panel DSNTIPP).
- DB2 cached the authorization information from a prior check.
- From a prior invocation of the RACF access control module, the routine had indicated that it should not be called again.
- DB2 GRANT statements.

Chapter 2. Planning

Implementing the RACF access control module involves the interaction of RACF, DB2 and z/OS system software, each with its own required skills. Therefore, it is important to understand the task at hand, organize the appropriate team members, and plan your implementation together.

This chapter provides the information you need to determine the tasks to be performed, identify the skills required, recognize decisions that should be made as a team, and understand how each choice you make affects DB2 authorization processing.

Mapping out the implementation tasks: A task roadmap

Table 1 shows the subtasks, participants, and associated procedures for implementing the RACF access control module.

Before you begin: Important decisions that you make during planning (Subtask 1) will be implemented during Subtasks 2–5.

Table 1. Task roadmap for implementing the RACF access control module

Subtask	Participants	Associated procedure
1. Plan your RACF access control module implementation.	DB2 administrator, RACF administrator, and MVS programmer	See Chapter 2, “Planning.”
2. Install and customize the RACF access control module.	MVS programmer	See Chapter 3, “Installing the RACF access control module,” on page 13.
3. (Optional) Define RACF classes for your DB2 resources, such as DB2 objects and administrative authorities.	MVS programmer	See Chapter 4, “Defining classes for the RACF access control module (optional),” on page 17.
4. Define RACF resources to protect your DB2 objects.	RACF administrator	See Chapter 5, “Protecting DB2 objects,” on page 23.
5. Define RACF resources to protect the DB2 administrative authorities.	RACF administrator	See Chapter 6, “Protecting DB2 administrative authorities,” on page 27.
6. Activate the RACF classes for your DB2 resources and administrative authorities.	RACF administrator	See Chapter 7, “Making your new RACF resources effective,” on page 29.
7. Restart the DB2 subsystem.	DB2 administrator	—

Identifying skill requirements

Organizing your team involves incorporating a variety of skill sets and may require you to include people from different disciplines if you work in a large organization. These skills are identified in terms of the roles or job titles of the people who specialize in those skills. For example, a task requiring MVS system skills is referred to as a task for the MVS programmer. If some of your team members have multiple skills, you may require fewer individuals to complete your team.

Your team for planning and implementing the RACF access control module should include the following members:

- MVS programmer
- RACF administrator

- DB2 administrator.

The following table lists the team members, tasks, and required skills for planning and implementing the RACF access control module.

Table 2. Roles, tasks, and skills for the implementation team

Role	Tasks	Required skills	Useful references
MVS programmer	• Install (customize, assemble, and link-edit) the RACF access control module • Define the RACF classes for use with DB2	• TSO skills • JCL knowledge • Assembler programming	• *z/OS Security Server RACF Macros and Interfaces* • *z/OS Security Server RACF System Programmer's Guide* • *DB2 Installation Guide* • *DB2 Administration Guide* • (optional) *z/OS Planning for Multilevel Security and the Common Criteria*
RACF administrator	• Plan RACF classes for use with DB2 • Define RACF resources to protect DB2 objects and administrative authorities • Activate the RACF classes for DB2	• RACF administration • RACF commands, such as the following: – ADDGROUP – ADDUSER – RALTER – RDEFINE – PERMIT – SETROPTS • TSO skills	• *z/OS Security Server RACF Security Administrator's Guide* • *z/OS Security Server RACF Command Language Reference* • (optional) *z/OS Planning for Multilevel Security and the Common Criteria*
DB2 administrator	• Plan the DB2 objects and administrative authorities to protect • Restart the DB2 subsystem	• DB2 basic operations • DB2 commands and authorization requirements • System and basic database administration	• *DB2 Administration Guide* • *DB2 SQL Reference* • *DB2 Data Sharing: Planning and Administration*

Planning for migration

This topic describes the considerations related to two types of migrations that you may encounter when installing the RACF access control module (supplied in the DSNXRXAC member of *prefix*.SDSNSAMP beginning in DB2 Version 8. One type of migration involves migrating from DB2 internal security where you do not use RACF for access control authorization to DB2 resources. The other involves migrating from a previous level of the DB2 access control module (supplied in the IRR@XACS member of SYS1.SAMPLIB) where you are already using RACF for access control authorization to DB2 resources.

Migrating from DB2 internal security

When migrating from DB2 internal security to the RACF access control module, you need not migrate protection of all DB2 objects at once. You can begin using the RACF access control module before defining profiles to protect all DB2 object types. Consider adding the WARNING option of RDEFINE and RALTER commands when you protect DB2 objects. The use of warnings might ease your migration by allowing you to see ICH408I messages that identify profiles that would fail a request.

Any request to access a DB2 object protected by a RACF profile with the WARNING option is always allowed. If the request would have failed without the WARNING option, an ICH408I message is generated to identify the first profile (in the sequence of RACF authorization checking) that would have failed the request.

Note: When the WARNING option is added to a resource requested by a user with a DB2 administrative authority, such as SYSADM, DBADM, or in some cases SYSCTRL, that would also allow the user to access the object, you can ignore the warning message.

If the RACF access control module determines that there is no administrative authority profile and no profile to protect a particular DB2 object (or the class corresponding to a particular DB2 resource is not active), it defers to DB2 for authority checking.

For example, suppose only the set of RACF profiles to protect DB2 tables has been defined and the classes for all other object types have not been made active. In this case, the RACF access control module performs profile checking for DB2 tables, views, and indexes and it defers to DB2 for authority checking of other object types, such as plans, packages, and databases.

Guideline: All DB2 administrative authorities should be defined with UACC(NONE) before you activate the RACF access control module. You can then selectively authorize specific users at a higher level by executing the PERMIT command.

Migrating from the IRR@XACS module

If your installation currently uses a previous level of the DB2 access control module (supplied in the IRR@XACS member of SYS1.SAMPLIB), you can migrate to module DSNXRXAC (supplied in the DSNXRXAC member of *prefix*.SDSNSAMP beginning in DB2 Version 8) while sharing the same RACF database.

If you used warning mode to allow access to DB2 objects prior to DB2 V8, the IRR@XACS module suppressed the ICH408I messages that identified the profiles that would have failed requests if warning mode had not been active but to which the requested access was allowed. The DSNXRXAC module does not suppress the messages.

Sharing the RACF database

You can share the RACF database with both a DB2 Version 7 subsystem (with module IRR@XACS) and a DB2 Version 8 subsystem (with module DSNXRXAC). The important action is that you must correctly associate each module with the correct DB2 version. Both DSNXRXAC and IRR@XACS contain support to prevent them from being invoked by the incorrect version of DB2 when you install APAR OA05967 on your z/OS V1R3, V1R4, and V1R5 systems. Systems running z/OS V1R6, and later releases, contain this support and, therefore, APAR OA05967 is not applicable.

Choosing the RACF access control module customization options

This section describes the customization options and corresponding class name formats related to the RACF access control module. Customizing the RACF access control module is optional. It is required *only* when you do not use the default values.

Using the default values allows the RACF access control module to use the classes in the class descriptor table (CDT) supplied by IBM®. (See Appendix B, “Supplied RACF resource classes for DB2,” on page 59.) When you modify the customization options from their default values, you might need to define classes in the installation-supplied class descriptor table.

The RACF access control module uses the values `&CLASSOPT`, `&CLASSNMT`, and `&CHAROPT` to determine the format of the class names and resource names it will construct to protect the DB2 objects. The decisions you make about changing or keeping these defaults should be well understood before you complete “Steps for installing the RACF access control module” on page 13.

Restriction: Each option that you specify in the RACF access control module applies to the entire DB2 subsystem using the module. This means that the `&CLASSOPT`, `&CLASSNMT`, and `&CHAROPT` values you select apply to all classes used by that DB2 subsystem. If you have multiple DB2 subsystems and need to apply different values across subsystems, install the RACF access control module separately on each subsystem, each with its own set of processing options.

Table 3. Set symbols and values

Set symbol	Description	Default value	See...
`&CLASSOPT`	Specifies the class scope option. **Valid values:** **1** Single-subsystem scope **2** Multiple-subsystem scope	**2**	“Choosing the class scope” on page 9
`&CLASSNMT`	Specifies the class name *root*, which is characters 2–5 of the class name, and is used only when you also specify `&CLASSOPT 2`. (When you specify `&CLASSOPT 1`, the DB2 subsystem name or, if data sharing, the DB2 group attachment name, is used as the class name root.) **Rule:** This value must be 1–4 characters long.	**DSN**	“Choosing the class name root and suffix” on page 10
`&CHAROPT`	Specifies the class name *suffix*, which is the last character of the class name for installation-defined classes. **Valid values:** 0–9, #, @, $, or a blank character (' ').	**1**	“Choosing the class name root and suffix” on page 10
`&ERROROPT`	Specifies the action to take in the event of an initialization or authorization error. **Valid values:** **1** Native DB2 authorization is used. This is the default. **2** The DB2 subsystem is requested to stop.	**1**	“Choosing the error option” on page 10
`&PCELLCT`	Specifies the number of primary work area cells	**50**	“Customizing the number of exit work area cells” on page 10
`&SCELLCT`	Specifies the number of secondary work area cells	**50**	“Customizing the number of exit work area cells” on page 10
`&SERVICELEVEL`	For IBM use only		

The default values for all customization options as shipped with the RACF access control module are shown in Figure 1 on page 9.

```
       GBLC  &CLASSNMT,&CHAROPT,&CLASSOPT
       GBLA  &PCELLCT,&SCELLCT
&CLASSOPT     SETC  '2'     1 - Use Single Subsystem Class Scope
*                               Classification Model I
*                               (One set of classes for EACH subsys)
*                           2 - Use Multi-Subsystem Class Scope
*                               Classification Model II
*                               (One set of classes for ALL subsys)
&CLASSNMT     SETC  'DSN'   DB2 Subsystem Name (Up to 4 chars)
&CHAROPT      SETC  '1'     One character suffix (0-9, #, @ or $)
&ERROROPT     SETC  '1'     1 - Use Native DB2 authorization
*                           2 - Stop the DB2 subsystem
&PCELLCT      SETA  50      Primary Cell Count
&SCELLCT      SETA  50      Secondary Cell Count
```

Figure 1. Default values for installation options

Choosing the class scope

The system programmer can alter the `&CLASSOPT` field of the modifiable assembler source statement in the RACF access control module to select the desired scope for the DB2 classes that will protect DB2 objects and privileges.

`&CLASSOPT` value	Scope	Classification model
1	Single-subsystem scope	1
2	Multiple-subsystem scope **Note:** *This is the default.*	2

When you select *single-subsystem scope*, you are choosing to define a separate set of classes for each DB2 subsystem that uses the RACF access control module. In general, you cannot use the classes in the supplied class descriptor table (ICHRRCDX) in single-subsystem scope.

When you select the *multiple-subsystem scope*, you are choosing to share a set of classes across all DB2 subsystems using RACF access control module, rather than defining a separate set for each. In multiple-subsystem scope, you can use the classes in the supplied class descriptor table (ICHRRCDX). This scope generally requires less administrative effort to set up and is the scope that most installations choose.

One general resource class is associated with each DB2 object type. You can define up to two classes for each object type and set them up as associated members or grouping classes. The list of supported DB2 objects and class abbreviations is defined in Table 4 on page 24. If only one class is used for an object, it must be defined with the member prefix. An additional class is used to support DB2 administrative authorities. The format of the class names of DB2 objects depends on the classification model you use.

System considerations

When you choose single-subsystem scope and need to add a new DB2 subsystem
or upgrade the RACF access control module to support a new DB2 object type, you
must add new RACF classes in the RACF class descriptor table. If you are running
z/OS Version 1 Release 6 or later, you can add RACF classes without a system

#
#
#

restart. If you are running z/OS Version 1 Release 5 or earlier, you must IPL your system to install the new class entries in the RACF class descriptor table and the RACF router table.

When you choose multiple-subsystem scope, you can avoid the need to IPL because you might not need to install new RACF classes. You can dynamically define new RACF resources to protect DB2 objects using existing classes. See *z/OS Security Server RACF Security Administrator's Guide* for details about defining and activating protection for RACF resources.

If you define new RACF resources to protect DB2 objects in a class that was not active at the time your DB2 subsystem was started, you need to restart the DB2 subsystem to activate the new resources. If the class was active at startup time, then you can dynamically activate the new resources using the TSO **SETROPTS RACLIST REFRESH** command for the class. See Chapter 7, "Making your new RACF resources effective," on page 29.

Choosing the class name root and suffix

Once a class scope is selected, the system programmer can use the `&CHAROPT` and `&CLASSNMT` SET symbols to alter the default naming conventions for the resource classes and profiles you use to protect DB2 objects and administrative authorities.

Choosing the error option

Set the `&ERROROPT` value to choose which action you want the system to take in the event of an initialization or authorization error. If you do not set this option or allow it to default to `&ERROROPT 1`, native DB2 authorization is used in the event of an error.

If you select `&ERROROPT 2`, you can request the DB2 subsystem to stop when one of the following events occurs:

- An initialization error, such as when there are no active RACF classes found for the initializing DB2 subsystem.
- The exit routine abends, causing the accumulated number of exit routine abends to exceed the threshold specified during installation (`AUTH EXIT LIMIT`).
- DB2 receives an unexpected return code (EXPLRC1) from the RACF access control module.

Customizing the number of exit work area cells

When you invoke the RACF access control module, it uses CPOOL cells as a work area to contain local variables. When you invoke the RACF access control module for initialization, it allocates a primary pool of work area cells to be used on authorization requests. Each time the RACF access control module is invoked for an authorization request, it obtains a cell and returns it when processing completes. If there are no more cells available, it uses a secondary pool of cells. You can control the number of cells allocated in the primary and secondary cell pools with the `&PCELLCT` and `&SCELLCT` SET symbols.

Guideline: Use the `&PCELLCT` and `&SCELLCT` default values.

Planning RACF security for DB2

The most significant part of the planning process is planning to expand RACF protection and administration to DB2 subsystem resources. Plan to cover the following tasks.

1. Examining the current RACF environment, including the user group structure, resource naming conventions, and use of grouping classes.
2. Examining the DB2 objects, looking for naming conventions and other similarities in resource names that you can exploit with generic RACF profiles.
3. Examining the GRANT authorizations in place for DB2 objects to see which RACF user groups you can define, or exploit, to reduce the RACF authorizations you must create using the RACF PERMIT command.
4. Planning which DB2 objects and administrative authorities to protect, determining access requirements, and incorporating the new subsystem resources into the current RACF structure.
5. Considering the use of RACF variables to facilitate resource naming conventions for DB2 resources.
6. Integrating new DB2 users into the RACF user structure and delegating RACF group and class authorities.

Chapter 3. Installing the RACF access control module

The RACF access control module is an assembler source module that resides in the DSNXRXAC member of the *prefix*.SDSNSAMP library. Before your installation can use RACF to protect DB2 objects and authorities, you need to install the RACF access control module. To install the RACF access control module for a DB2 subsystem, you will copy, customize as needed, assemble, and link-edit the module into the DB2 exit library (*prefix*.SDSNEXIT).

You can modify the way the RACF access control module works by customizing several assembler SET symbols located in the top of the source data set. The default values for all customization options as shipped with the RACF access control module are shown in Figure 1 on page 9. For details about deciding to keep or change the default values, see "Choosing the RACF access control module customization options" on page 7.

Multiple DB2 subsystems can share the same copy of the RACF access control module as long as they use the same customization options. When subsystems require different options, you must install additional copies of the RACF access control module.

After you install the RACF access control module, it will become active the next time the DB2 subsystem is restarted when at least one RACF class associated with the DB2 subsystem is active at the time of the restart. Before restarting DB2, be sure that your implementation team has already defined appropriate RACF resources in the active DB2 classes or else your installation might cause unintended DB2 authorization failures or exposures.

Steps for installing the RACF access control module

Before you begin: You must have MVS system programming skills to complete this procedure. In Step 3, you can optionally customize the RACF access control module to modify several important authorization processing options. For details, see "Choosing the RACF access control module customization options" on page 7. Be sure to consult your implementation team to find out which customization options are needed, if any. In addition, you may wish to have *DB2 Installation Guide* available as a reference.

1. Verify that you have the current version of the RACF access control module,
including all required maintenance, such as APAR OA05967 for z/OS Version 1
Release 3, z/OS Version 1 Release 4, and z/OS Version 1 Release 5
installations. (APAR OA05967 is not applicable for z/OS Version 1 Release 6
and later releases.)

2. Locate the DSNXRXAC member (containing the RACF access control module) in the *prefix*.SDSNSAMP library and copy it to a private library.

   ____

3. Optionally, customize your private copy of the RACF access control module by modifying the assembler SET options from their default values. The options you use in this step will affect DB2 authorization processing so use the values chosen by your implementation team. See "Choosing the RACF access control module customization options" on page 7.

   ____

4. Use the DB2 installation job DSNTIJEX to assemble and link-edit the RACF access control module into the APF-authorized DB2 exit load library (*prefix*.SDSNEXIT). If you use another target library, you might have to change the STEPLIB or JOBLIB concatenations in the DB2 startup procedures. For information about using the DB2 installation jobs, see *DB2 Installation Guide*.
   a. Modify Step 3 (JEX0003) of DSNTIJEX to point to the library containing your customized version of DSNXRXAC and then run it.

   ---

   b. If you have two or more DB2 subsystems and you want to use different assembler SET options for each subsystem (or you just want to have separate exit load libraries), repeat Step 4a for each DB2 subsystem.

After you complete these steps, the RACF access control module will be initialized the next time the DB2 subsystem is started. The initialization function will succeed and the RACF access control module will become active only if DB2 resource classes are active at the time of the restart. If the RACF access control module is active, DB2 will invoke RACF for authority checking.

You can determine whether the DB2 will perform DB2 authorization checks by reviewing the IRR9*nnx* messages and any DSNX210I message you receive during DB2 initialization.

Guideline: If you receive the IRR912I message during initialization, your exit routine is not active and native DB2 authorization checking will be used.

Steps for testing that your exit routine is active

You can perform the following steps to cause an authorization failure to test if your exit routine is active:

1. Choose a RACF-defined DB2 table on which to execute a SELECT statement
and choose an authorization ID to perform the SELECT. The authorization ID
must *not* own the table and have *none* of the following access authorizations:
• DB2 administrative authority (installation SYSADM, SYSADM, SYSCTRL, or
DBADM for the database containing the table)
• DB2 SELECT privilege on the chosen table
• RACF authorization for the SELECT privilege on the chosen table
• RACF authorization for READ access to the chosen table.
#

---

2. Use the authorization ID to execute a SELECT statement on the table. The SELECT statement should fail.

---

3. Review the resulting ICH408I information messages related to DB2 resources and examine the RACF return code.

When you complete this test, you will know if RACF is performing DB2 authorization checking. If it is, the RACF access control module is active. Also, you might check the DB2 trace facility. The DB2 trace record IFCID 314 is only generated when the RACF access control module is active.

Using RACF informational messages

Once you successfully activate the RACF access control module and DB2 invokes RACF for authorization checking, you can use the information found in messages IRR908I through IRR911I to see how it is set up for a particular subsystem. These messages identify the:

- Version and length of the RACF access control module
- DB2 subsystem name or, if data sharing, the DB2 group attachment name
- DB2 FMID (for example, HDRE810 for DB2 Version 8) or APAR number associated with the module
- Options used for the module

  For example, `&ERROROPT` specifies the correct action to be taken for DB2 initialization and authorization errors.

  Note: The MVS programmer sets these options. For detailed information, see "Choosing the RACF access control module customization options" on page 7.
- Classes that the module is trying to use
- Classes for which a RACROUTE request was successful.

These messages are routed only to the system log and occur only at DB2 initialization time, not during authorization checking. Therefore, these messages are issued regardless of whether any authorization checks have been made, are issued even when DB2 initialization fails.

Chapter 4. Defining classes for the RACF access control module (optional)

Defining classes for the RACF access control module is optional. It is required *only* when you do not use the defaults.

When you change the `&CLASSOPT` or `&CLASSNMT` assembler SET symbols from their default values, you need to define your own classes in the class descriptor table (CDT).

Useful references:
- If you are running z/OS Version 1 Release 6 or later, define the classes in the dynamic class descriptor table so that you do not need to re-IPL to activate the new classes. For information about the dynamic class descriptor table, see *z/OS Security Server RACF Security Administrator's Guide*.
- If you are running z/OS Version 1 Release 5 or earlier, you must define your own classes in the installation-defined CDT, define corresponding entries in the RACF router table, and re-IPL to activate the new classes. For details on defining CDT and router table entries, see *z/OS Security Server RACF System Programmer's Guide*.

It is not necessary to define classes for DB2 objects and administrative authorities that are not protected by the RACF access control module. To see which DB2 classes are protected, see Appendix B, "Supplied RACF resource classes for DB2," on page 59.

You can define classes for DB2 objects and you can define classes for administrative authorities. See the formats for these class names in:
- "Defining class names for DB2 objects" on page 18
- "Defining class names for administrative authorities" on page 20

When using the single-subsystem scope, the RACF access control module builds class names dynamically by concatenating the DB2 subsystem name, or group attachment name, with the object type. As a result, multiple DB2 subsystems can use the same copy of the RACF access control module. However, you must create an installation-defined set of classes for each subsystem.

When using the multiple-subsystem scope, the RACF access control module builds class names dynamically by concatenating the `&CLASSNMT` with the object type. As a result, any DB2 subsystem with the same `&CLASSNMT` can use the same copy of the RACF access control module. You may create an installation-defined set of classes for each subsystem or you may choose to use the supplied classes instead.

Restrictions:
1. If you choose to use installation-defined classes, you must use installation-defined classes with all objects for the same copy of the RACF access control module. You cannot mix classes supplied by IBM and installation-defined classes. To use both types, you must use different versions of the RACF access control module.
2. RACF expects that installation-defined classes have the same class descriptor table attributes as the corresponding DB2 classes supplied by IBM. (For a list of these attributes, see *z/OS Security Server RACF Macros and Interfaces*.)

Defining class names for DB2 objects

In the supplied class descriptor table (ICHRRCDX), two classes are defined for each DB2 object type (except for the DB2 view object which shares classes with the table object), so that each object type has an associated member class and an associated grouping class. See Appendix B, “Supplied RACF resource classes for DB2,” on page 59 for a list of the supplied RACF classes associated with each DB2 object type. (For general information about using member and grouping RACF classes, see *z/OS Security Server RACF Security Administrator’s Guide*.) Table 4 on page 24 lists the supported DB2 objects and class abbreviations.

Installations defining their own classes can also define two classes for each object type if member and grouping classes are desired. If only one class is defined for each object type, the class name must begin with `M` (*not* `G`).

The actual format of the class names of DB2 objects depends on the classification model being used.

Note: The default name for the DB2 administrative authorities class is DSNADM. You can define an additional RACF class. See Chapter 6, “Protecting DB2 administrative authorities,” on page 27.

Defining class names for DB2 objects in single-subsystem scope

When you select this model, the RACF access control module inserts the DB2 subsystem name, or group attachment name, when it constructs RACF class names. The classes that you define must follow this format:

`ayyyyxxz`

where:

a	is `M` for member class or `G` for grouping class
yyyy	is the DB2 subsystem name or, if data sharing, the DB2 group attachment name (from XAPLGPAT)
xx	is the type of DB2 object (See Table 4 on page 24 for valid values for each DB2 object.)
z	is the `&CHAROPT` value (The default is `1`.)

In single-subsystem scope, the class names of the DB2 object classes contain the DB2 subsystem name or DB2 group attachment name but the profile names of resources in those classes do not. Therefore, in single-subsystem scope, you must define a separate class name for each subsystem that will use the RACF access control module. See Figure 2 on page 19.

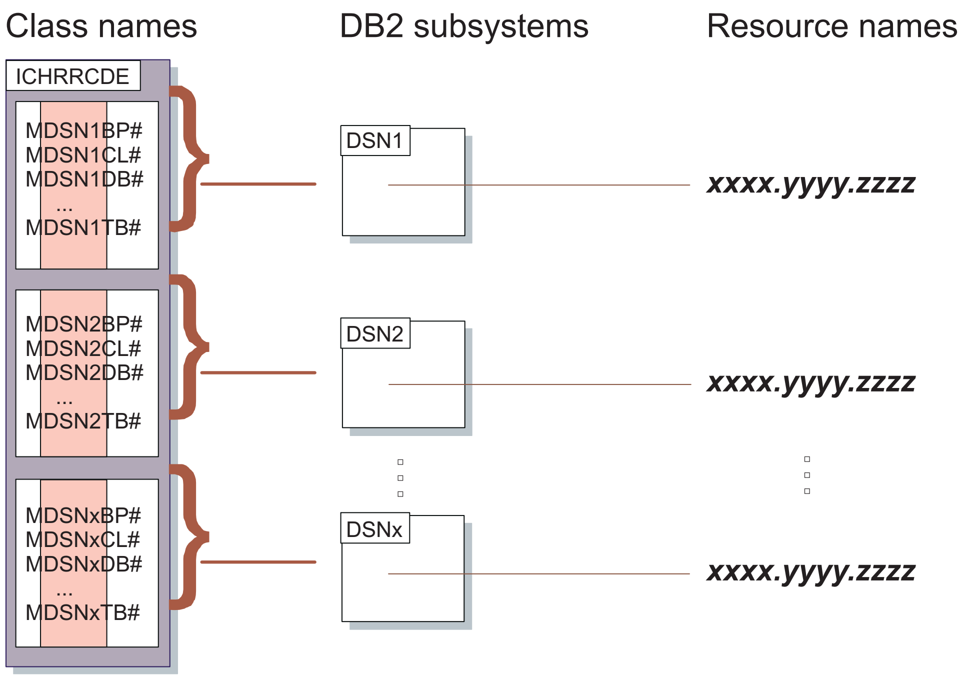

Figure 2. Single-subsystem scope classes

When you use the single-subsystem scope, you cannot use the classes provided in the supplied class descriptor table (ICHRRCDX) unless you are using the default DB2 subsystem name `DSN` and have altered the `&CHAROPT` variable in the RACF access control module to be a blank character (' '). However, in single-subsystem scope, you must still define a separate class name for every other subsystem that will share the RACF access control module.

When you define your own classes, you can define two classes for each object type if member and grouping classes are desired. If only one class is defined for each object type, the class name must begin with `M` (*not* `G`).

To locate information about defining your own classes in the CDT, see "**Useful**
references" at the beginning of Chapter 4, "Defining classes for the RACF access
control module (optional)," on page 17.

Defining class names for DB2 objects in multiple-subsystem scope

When you select this model, the RACF access control module places the DB2 subsystem name in the resource name. Class names that you define must have this format:

abbbbxxz

where:

a	Is `M` for member class or `G` for grouping class
bbbb	Is the `&CLASSNMT` value (the default value is `DSN`)
xx	Is the type of DB2 object (see Table 4 on page 24 for valid values)
z	Is the `&CHAROPT` value (ignored if `&CLASSNMT='DSN'`)

In multiple-subsystem scope, profile names of resources in the DB2 object classes are prefixed with the DB2 subsystem name, or group attachment name, but the class names are not. See Figure 3.

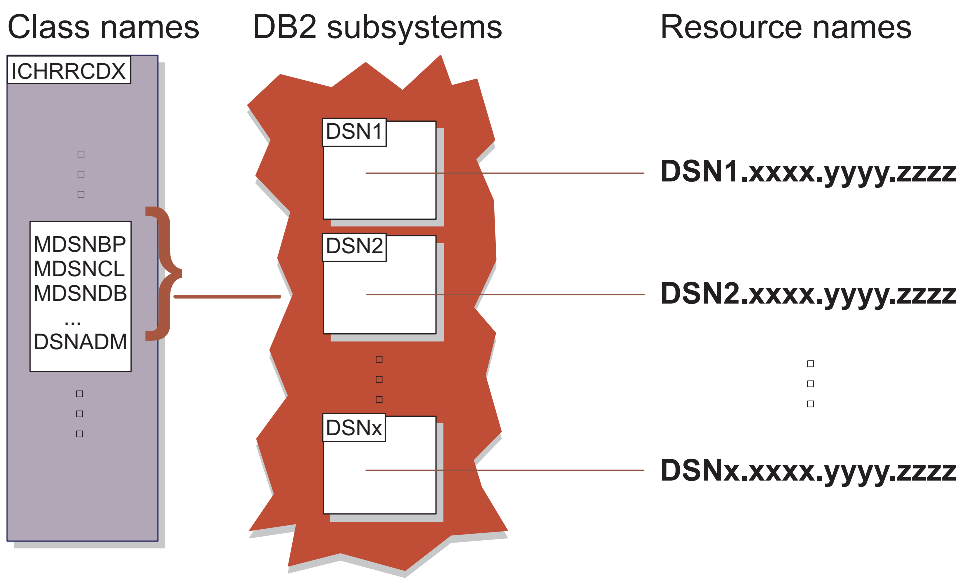

Figure 3. Multiple-subsystem scope classes

If you use the multiple-subsystem scope and the default `&CLASSNMT` value (`DSN`), you can use the classes provided in the supplied class descriptor table (ICHRRCDX). Any subsystem sharing the RACF access control module can share the same set of classes. You do not need to define a separate set of classes for each subsystem.

You can change `&CLASSNMT` if you do not want to use the default (`DSN`) value. However, if you set `&CLASSNMT` to a value other than `DSN`, you must define classes in the class descriptor table (CDT). You can define two classes for each object type if both member and grouping classes are desired. If only one class is defined for each object type, the class name must begin with `M` (*not* `G`).

To locate information about defining your own classes in the CDT, see "**Useful**
references" at the beginning of Chapter 4, "Defining classes for the RACF access
control module (optional)," on page 17.

Defining class names for administrative authorities

The DB2 administrative authority class (named DSNADM, by default) allows RACF security administrators to create profiles that are suffixed with specific DB2 administrative authorities, to allow users to access certain resources for specified DB2 subsystems or groups. The format is dependent on the scope (`&CLASSOPT`) specified.

Defining class names for DB2 administrative authorities in single-subsystem scope

When you select `&CLASSOPT 1`, the RACF access control module places the DB2 subsystem name, or group attachment name, in the administrative authority class name. Define administrative authority class names in single-subsystem scope using this format:

*yyyy***ADM***z*

where:

yyyy	Is the DB2 subsystem name or, if data sharing, the DB2 group attachment name (from XAPLGPAT)
ADM	Is the designation for administrative authority classes
z	Is the `&CHAROPT` value (the default value is `1`)

In single-subsystem scope, the class names of the DB2 administrative authority classes contain the DB2 subsystem name, or DB2 group attachment name, but the profile names of resources in those classes do not. Therefore, in single-subsystem scope, you must define a separate class name for each subsystem that will use the RACF access control module.

When you select single-subsystem scope, you cannot use the DB2 administrative authority class called DSNADM that is provided in the supplied class descriptor table (ICHRRCDX). You must define your own class in the class descriptor table (CDT), unless you use the default DB2 subsystem name `DSN` and have altered the `&CHAROPT` variable in the RACF access control module to be a blank character (' '). However, in single-subsystem scope, you must still define a separate class name for every other subsystem that will share the RACF access control module.

To locate information about defining your own classes in the CDT, see "**Useful**
references" at the beginning of Chapter 4, "Defining classes for the RACF access
control module (optional)," on page 17.

Defining class names for DB2 administrative authorities in multiple-subsystem scope

When you select `&CLASSOPT 2` or allow it to default, the RACF access control module does not use the DB2 subsystem name or group attachment name in the class name for administrative authorities. Define administrative authority class names in multiple-subsystem scope using this format:

*yyyy***ADM***z*

where:

yyyy	Is the `&CLASSNMT` value (the default value is `DSN`)
ADM	Is the designation for administrative authority classes
z	Is the `&CHAROPT` value, which is ignored if `&CLASSNMT` is set to `DSN`

In multiple-subsystem scope, profile names of resources in the DB2 administrative authority class are prefixed with the DB2 subsystem name, or DB2 group attachment name, but the class name does not. Therefore, installations using multiple-subsystem scope and the default `&CLASSNMT` value (`DSN`) can use the default DB2 administrative authority class (DSNADM) provided in the supplied class descriptor table (ICHRRCDX). Any subsystem sharing the RACF access control module can share the same class. A separate class does not need to be defined for each DB2 subsystem.

If you set `&CLASSNMT` to a value other than `DSN`, you must define a DB2 administrative authority class in the class descriptor table (CDT).

To locate information about defining your own classes in the CDT, see "**Useful**
references" at the beginning of Chapter 4, "Defining classes for the RACF access
control module (optional)," on page 17.

Chapter 5. Protecting DB2 objects

The resources that apply to a particular invocation of the RACF access control module depend on the input object type (XAPLTYPE) and the privilege being checked (XAPLPRIV). The object types and the names of their associated privileges are shown in Appendix D, “RACF authorization checking reference,” on page 67. See the DB2 macro DSNXAPRV in *prefix*.SDSNMACS to find the numeric XAPLPRIV values (used by the RACF access control module) that correspond to the privilege names.

The RACF access control module constructs general resource class and profile names for DB2 objects based on the options you specified using the assembler SET symbols:

SET symbol	Default value	Description
`&CLASSOPT`	2	Specifies the classification model
`&CLASSNMT`	DSN	Specifies the class name root
`&CHAROPT`	1	Specifies the class name suffix

The `&CLASSOPT`, `&CLASSNMT`, and `&CHAROPT` options specify the format of the class names and resource profile names used by the RACF access control module. These options are global for each DB2 subsystem, and must be the same for all classes. Each instance of the RACF access control module can only be set up to process one classification model or the other, but not both. See “Choosing the RACF access control module customization options” on page 7 for more information.

If your installation is using the default values for these options, you can use the classes in the supplied class descriptor table (ICHRRCDX). Additional classes do not need to be defined.

Security administrators must define the RACF resources to protect DB2 objects using names that correspond to the format required by the options set in the RACF access control module. The formats for the resource profile names are described in “Defining resource names for DB2 objects” on page 24.

DB2 object types

Each authorization request has an associated DB2 object type. DB2 provides the object type as a 1-character abbreviation in the XAPLTYPE field. This abbreviation is used by the RACF access control module in conjunction with the code for the requested privilege (see *DB2 Administration Guide*) to determine which checking to perform.

A non-valid XAPLTYPE or XAPLPRIV passed to the RACF access control module during authorization checking will cause the RACF access control module to return a return code of 4 (“RACF access not determined; perform DB2 access checking”).

Table 4 on page 24 lists the DB2 objects, the DB2 abbreviations used in the XAPL, and the abbreviations used in the RACF general resource grouping and member class names (GDSN*xx* and MDSN*xx*):

Table 4. DB2 object abbreviations

DB2 object	DB2 object abbreviation	RACF class abbreviation
Buffer pool	B	BP
Collection	C	CL
Database	D	DB
Java™ archive (JAR)	J	JR
Package	K	PK
Plan	P	PN
Schema	M	SC
Sequence	Q	SQ
Storage group	S	SG
Stored procedure	O	SP
System	U	SM
Table or index	T	TB
Table space	R	TS
User-defined distinct type	E	UT
User-defined function	F	UF
View	V	TB

Defining resource names for DB2 objects

The RACF access control module builds resource names depending on the classification model being used.

For single-subsystem scope, the general format for resource name is:

```
[object-name.]privilege-name
```

For multiple-subsystem scope, the general format for resource name is:

```
DB2-subsystem.[object-name.]privilege-name

or, if data sharing:

DB2-group-attachment-name.[object-name.]privilege-name
```

For multiple-subsystem scope, the RACF access control module obtains the DB2 subsystem name, or group attachment name, from XAPLGPAT.

The RACF access control module uses resource names that are based on the object names and the associated privilege names. See “DB2 object types and object names” on page 25 and “Privilege names” on page 26.

Using generic RACF profiles

You can define a RACF resource that protects one or more DB2 objects that have the same security requirements by using generic RACF profiles. Using generic profiles allows you to exploit naming conventions and greatly reduce the number of RACF profiles you must define. Most generic profiles contain one or more masking characters to replace one or more characters or qualifiers of a resource name. See *z/OS Security Server RACF Security Administrator’s Guide* for complete details.

DB2 object types and object names

The RACF access control module constructs the RACF resources name using information passed in various fields (XAPLOBJN, XAPLOWNQ, XAPLREL1, and XAPLREL2). The content of these fields depends on the input object type, XAPLTYPE.

Table 5 defines the object name qualifiers used in RACF resource names for each DB2 object type:

Table 5. DB2 object name qualifiers for RACF resources

DB2 object	Object name qualifiers
buffer pool	*bufferpool-name*
collection	*collection-ID*
database	*database-name*
Java archive (JAR)	*schema-name.JAR-name*
package	*collection-ID.package-ID* *collection-ID* *owner*
plan	*plan-name* *owner*
schema	*schema-name* *schema-name.function-name* *schema-name.procedure-name* *schema-name.type-name*
sequence	*schema-name.sequence-name*
storage group	*storage-groupname*
stored procedure	*schema-name.procedure-name*
system	*owner* (BINDAGENT only)
table, index	*table-owner.table-name* *table-owner.table-name.column-name*
table space	*database-name.table-space-name*
user-defined distinct type	*schema-name.type-name*
user-defined function	*schema-name.function-name*
view	*view-owner.view-name* *table-owner.table-name* #

Note: The format of the DB2 object name qualifiers is defined by DB2. For more information, see *DB2 SQL Reference*.

Long object names

Some DB2 objects can have names containing up to 128 characters. Because RACF profile names are limited to 246 characters, the RACF access control module
# might truncate the schema name or table owner portion of the profile name to 100 characters when you use long object names. For example, consider the RACF profile name for the USAGE privilege on a JAR object:

```
DB2-subsystem.schema-name.JAR-name.USAGE
```

The schema name and JAR name can each contain a maximum of 128 characters. If the DB2 subsystem name is four characters, the length of the profile name would reach 268 characters and exceed the maximum name length unless the RACF access control module truncates schema name to 100 characters.

When you use long object names, consider schema name truncation to avoid unintended results, especially when you also use discrete RACF profiles. If truncation occurs, a single discrete profile might inadvertently protect multiple similarly named resources—if the first 100 characters of the schema names are the identical *and* the qualified object names, such as JAR name, subsystem name, and privilege name, are also identical.

Privilege names

The RACF access control module constructs the DB2 resource name using the DB2 privilege name as the lowest-level qualifier (RACF profile-name suffix) in the resource name. Each explicit privilege used as a low-level qualifier corresponds to one of the explicit privilege names that DB2 uses for a particular object. For a complete reference of all valid privilege names that can be used in a resource name for each DB2 object, see the tables in Appendix D, “RACF authorization checking reference,” on page 67.

Tip: You can authorize a user for one or more privileges on a DB2 object by defining a generic RACF profile using an asterisk (*****) in place of the privilege name and then permitting the user to the generic profile. However, if a more specific generic profile or a discrete profile also protect the same privilege or set of privileges, RACF will use those profiles to control access rather than the less specific generic profile.

See “DB2 GRANT statements” on page 46 for an example of using a generic character in place of the privilege name. (In contrast with SQL, in RACF a single asterisk (*****) matches characters within the scope of a single qualifier.)

Chapter 6. Protecting DB2 administrative authorities

The RACF access control module supports the DB2 concept of administrative authorities. DB2 administrative authorities often include privileges that are not explicit, have no name, and cannot be specifically granted. For example, the ability to terminate any utility job is included in the SYSOPR authority.

During authorization checking, if a user is not permitted access to the object through the object's resource profile, subsequent checks are made to determine if the user has been permitted access to system resources through their administrative authorities. These checks are made using profiles in the DB2 administrative authority class. The default name of this class is DSNADM.

DB2 Administration Guide documents the set of privileges that each DB2 administrative authority provides. The administrative authorities that apply to a particular invocation of the RACF access control module, depend on the input object type (XAPLTYPE) and the privilege being checked (XAPLPRIV). They are detailed in Appendix D, "RACF authorization checking reference," on page 67.

Like the names used to protect DB2 objects, the general resource class and profile names used to protect DB2 administrative authorities depend on the options specified with the assembler SET symbols.

Defining resource names for administrative authorities

The RACF access control module builds the resource names for administrative authorities based onthe classification model you selected.

For single-subsystem scope, the format for DB2 administrative authority resources is:

```
[object-name.]authority-name
```

For multiple-subsystem scope, the general format is:

```
DB2-subsystem.[object-name.]authority-name
```

or, if data sharing,

```
DB2-group-attachment-name.[object-name.]authority-name
```

For multiple-subsystem scope, the DB2 subsystem name or DB2 group attachment name is obtained from XAPLGPAT. The object name used depends on the DB2 administrative authority. See Table 6 on page 28.

DB2 administrative authorities and object names

The RACF access control module constructs the RACF resource name using information passed in various fields (XAPLOBJN, XAPLOWNQ, XAPLREL1, or XAPLREL2). The content of these fields depends on the input object type, XAPLTYPE.

Table 6 on page 28 lists the DB2 administrative authorities and the associated RACF object qualifiers:

Table 6. DB2 administrative authorities and object qualifiers

Administrative authority	**RACF object qualifier**
DBADM	*database-name*
DBCTRL	*database-name*
DBMAINT	*database-name*
PACKADM	*collection-ID*
SYSADM	—
SYSCTRL	—
SYSOPR	—

Note: The format of the DB2 object names is defined by DB2. For more information, see *DB2 SQL Reference*.

Chapter 7. Making your new RACF resources effective

If your DB2 subsystem was up and running when you defined your new DB2 objects and administrative authorities in Chapter 5, “Protecting DB2 objects,” on page 23 and Chapter 6, “Protecting DB2 administrative authorities,” on page 27, your new resource definitions are not in effect until you take explicit steps to make them effective. In order to be effective, the new RACF resource definitions must be read into storage for RACF access list checking.

#
#
#
Depending on whether the resource classes where you defined the new resources were active at the time your DB2 subsystem was started, you will execute different sets of commands to put your resource definitions in effect, as shown below.

If the class was not active

#
#
#
#
When you define new RACF resources to protect DB2 objects in a class that was not active at DB2 startup time, you need to stop the DB2 subsystem, activate the class, and then restart the DB2 subsystem to read the new profiles into storage and allow the new resource definitions to become effective.

Example:

From the MVS console, issue the following command:

```
–STOP DB2
```

Issue the following RACF commands:

```
# SETROPTS CLASSACT(classname)
```

From the MVS console, issue the following command:

```
–START DB2
```

If the class was active

#
#
#
#
When the class was active at DB2 startup time, you can dynamically refresh all the profiles in storage for this class and allow the new resource definitions to become effective by issuing the following RACF command. You do not need to restart the DB2 subsystem after you execute the RACLIST command.

Example:

Issue the following RACF command:

```
SETROPTS RACLIST(classname) REFRESH
```

Chapter 8. Debugging the RACF access control module

You can use IFCID 0314 to provide a trace record of the parameter list on return from the RACF access control module. Activate this trace by turning on performance trace class 22. See *DB2 Command Reference* for information about the DB2 performance trace.

You can correlate IFCID 0314 records and RACF SMF records by timestamp to determine which SMF record is associated with each IFCID record.

For more information about debugging the RACF access control module, see *DB2 Administration Guide*.

Dump titles for the RACF access control module

The RACF access control module generates the following dump titles:

```
COMPON=DB2,COMPID=5740DRE00,ISSUER=DSNX@FRR,MODULE=DSNX@XAC,
ABEND=S0sss,REASON=NONE    ,L=zzzzzzzz

COMPON=DB2,COMPID=5740DRE00,ISSUER=DSNX@FRR,MODULE=DSNX@XAC,
ABEND=S0sss,REASON=aaaaaaaa,L=zzzzzzzz

COMPON=DB2,COMPID=5740DRE00,ISSUER=DSNX@FRR,MODULE=DSNX@XAC,
ABEND=Uuuuu,REASON=NONE    ,L=zzzzzzzz

COMPON=DB2,COMPID=5740DRE00,ISSUER=DSNX@FRR,MODULE=DSNX@XAC,
ABEND=Uuuuu,REASON=aaaaaaaa,L=zzzzzzzz
```

where:

sss	is the system abend code
uuuu	is the user abend code
aaaaaaaa	is the abend reason code
zzzzzzzz	is the module length

Using the content of XAPLDIAG

The RACF access control module returns a parameter, XAPLDIAG, that DB2 and other program products can use to trap and obtain diagnostic information. When the RACF access control module issues the RACROUTE REQUEST=FASTAUTH macro for authorization checking, depending on the AUDIT options used with the check, the module can record the resulting SAF return code, RACF return code, and RACF reason code in XAPLDIAG. Each invocation of the RACF access control module can issue multiple RACROUTE REQUEST=FASTAUTH macros, but the module evaluates each return code generated and determines the single correct return code to send to DB2. (See “Authorization return and reason codes” on page 55.)

The RACF access control module can store up to 20 sets of return codes from RACROUTE REQUEST=FASTAUTH macros in XAPLDIAG, allowing the results of a specific RACROUTE REQUEST=FASTAUTH macro to be determined.

The XAPL parameter list can be captured using DB2 trace record IFCID 314. In addition, the return code and corresponding reason code (EXPLRC1 and

EXPLRC2) for authorization failures are captured in DB2 trace record IFCID 140. The DB2 trace facility is documented in *DB2 Command Reference*.

The content of XAPLDIAG depends on the return code and reason code from the RACF access control module.

- If EXPLRC1=4 and ECPLRC2=14 (decimal), the ALESERV failed and the module made no RACROUTE REQUEST=FASTAUTH checks. In this case the first word of XAPLDIAG contains the non-zero ALESERV return code.
- Otherwise, each word of XAPLDIAG can contain a SAF return code, RACF return code, and RACF reason code corresponding to a non-zero return code from a RACROUTE REQUEST=FASTAUTH macro. Information related to non-zero return codes is stored in XAPLDIAG beginning with the first word until information related to all non-zero return codes has been stored, or until the XAPLDIAG area has filled. XAPLDIAG contains 20 words, allowing information related to 20 FASTAUTH requests to be stored for an invocation of the RACF access control module. If more than 20 FASTAUTH requests are issued, only the first 20 sets of return codes are stored.

DBADM authorization checking for the CREATE VIEW privilege may result in more than 20 FASTAUTH requests because a CREATE VIEW request may reference tables, or a combination of tables and views, from multiple databases. DB2 passes the names of all the databases referenced in the CREATE VIEW using a database list pointed to by XAPLDBSP. If SYSCTRL or SYSADM authorization checking does not grant the CREATE VIEW privilege and the XAPLCRVW field indicates that DBACRVW is enabled, the RACF access control module checks the user's DBADM authorization for each database in the list. The result of each DBADM check is placed in the XAPLDBDA field associated with each database. The RACF access control module updates XAPLDBDA with the following codes:

Y	Access to the database is allowed.
N	Access to the database is not allowed.
U	RACF was unable to return a decision. This occurs when the FASTAUTH request returns a SAF return code of X'04'.

The database list pointed to by XAPLDBSP is made up of four-word database information structures mapped by the XAPLDBS macro.

```
XAPLDBNP DS F PTR TO NEXT DATABASE INFORMATION STRUCTURE
XAPLDBNM DS CL8 DATABASE NAME
XAPLDBDA DS CL1 'Y' - IS DBADM
XAPLRSV5 DS CL3 RESERVED - UNUSED
```

Although DBADM checks may be done for multiple databases, only the results of the first 20 FASTAUTH requests are stored in XAPLDIAG. The results of all DBADM checking for each database is contained in the XAPL parameter list and is available using DB2 trace record IFCID 314.

The RACF access control module truncates the SAF return codes and RACF return codes to one byte, and the RACF reason code to two bytes, before storing them in XAPLDIAG. The format of each word in XAPLDIAG is:

xxyyzzzz

where:

xx	is the 1-byte SAF return code
yy	is the 1-byte RACF return code
zzzz	is the 2-byte RACF reason code

For a list of the RACF return codes and reason codes and their meanings, see the RACROUTE REQUEST=FASTAUTH section of *z/OS Security Server RACROUTE Macro Reference*.

For additional information on common problems that can occur as a result of adding installation-defined classes to the class descriptor table (CDT) for DB2 objects, see “Common problems and considerations” on page 51.

Parameter list for the access control authorization routine

Figure 4 shows how the parameter list points to other information.

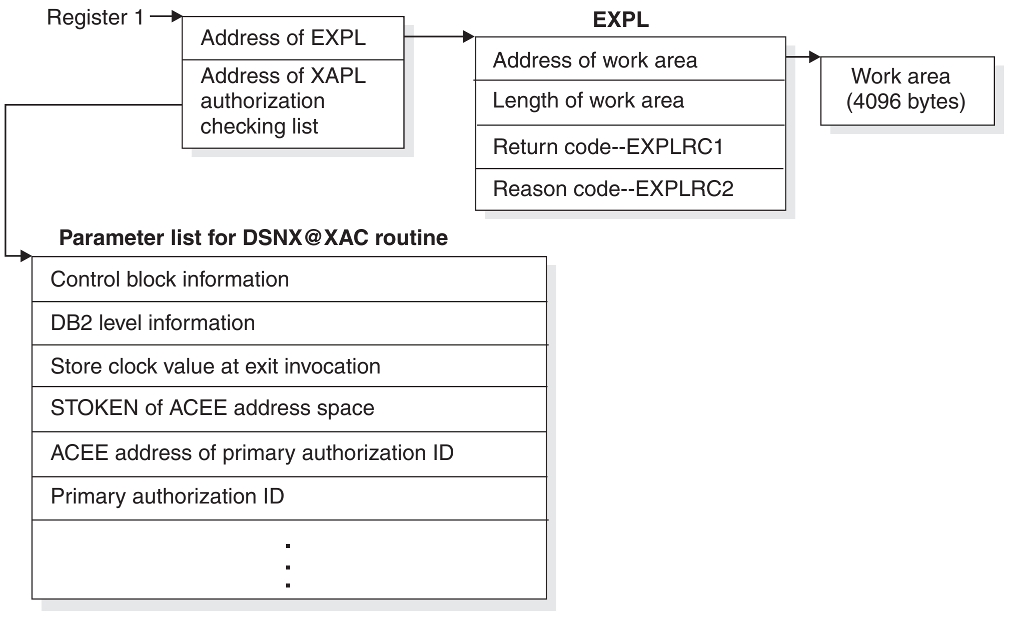
Register 1
Address of EXPL
Address of XAPL authorization checking list
EXPL
Address of work area
Length of work area
Return code--EXPLRC1
Reason code--EXPLRC2
Work area (4096 bytes)
Parameter list for DSNX@XAC routine
Control block information
DB2 level information
Store clock value at exit invocation
STOKEN of ACEE address space
ACEE address of primary authorization ID
Primary authorization ID
.
.
.

Figure 4. How an authorization routine's parameter list points to other information

The work area (4096 bytes) is obtained once during the startup of DB2 and only released when DB2 is shut down. The work area is shared by all invocations of the RACF access control module. See *DB2 Administration Guide* for exit-specific parameter information.

Chapter 9. Auditing for the RACF access control module

The RACF access control module allows you to use RACF resource profiles to check authorization for DB2 privileges and authorities. With these profiles, which represent the various DB2 privileges, you can use the RACF auditing tools to extract the information you need.

You can use the SMF data unload utility or the RACF report writer to extract and format the SMF records. When the RACF access control module uses a RACROUTE REQUEST=FASTAUTH request to create an audit record, the record contains log string data that includes additional diagnosis information described in "Using log string data" on page 36. You can use the log string information to link DB2 trace record IFCID 314 and a corresponding RACF SMF record.

In addition, you can use the RACF informational messages. For more information, see "Using RACF informational messages" on page 15.

Example of resource checking

The following example shows the series of RACF resources that are checked when a user issues the SELECT statement.

When RACF checks authorization, the requestor must own the object or have at least READ access to one of the following profiles:

Profile name	Class	Note
subsystem.table-name.table-owner.SELECT	MDSNTB	Gives access to the table
subsystem.database-name.DBADM	DSNADM	Gives access to the database that holds the table
subsystem.SYSCTRL	DSNADM	Bypassed for user tables
subsystem.SYSADM	DSNADM	—

RACF produces an SMF record for a failure only after checking the entire list of profiles and the requestor fails to meet any of the requirements. RACF does not produce an audit record if:

- The requestor meets any of the requirements and access is granted, or
- The RACF access control module returns the authority checking responsibility to DB2.

If DB2 objects were defined to RACF using the WARNING option, you will see ICH408I messages that identify those profiles that would fail a request and the requested access will be allowed.

Note: For DB2 releases prior to DB2 V8, the ICH408I messages were suppressed.

If the WARN option is added to a resource that is requested by a user with a DB2 administrative authority, such as SYSADM, DBADM or in some cases, SYSCTRL, that normally allows the user to access the object, the user can ignore the WARNING message.

An audit record is produced for the first resource that has auditing indicated by the covering profile and receives a return code of 8.

RACF produces an SMF record for a success when the requestor indicates that should be preformed.

For a list of the RACF classes, see Appendix B, "Supplied RACF resource classes for DB2," on page 59. For a full list of each RACF resource checked for each privilege, see Appendix D, "RACF authorization checking reference," on page 67.

Using log string data

The log string data consists of information that can help you audit DB2 successfully. DB2 uses the XAPL parameter list (DSNDXAPL macro) to pass log string information to the RACF access control module. The `LOGSTR=` parameter of the RACROUTE REQUEST=FASTAUTH request contains the input portion of XAPL and does the following:

- Identifies the RACF access control module request that caused RACF to create the audit record. The RACF profile causing the audit record to be cut could be a profile that provides a DB2 administrative authority and might not identify the specific DB2 resource being accessed. The log string data contains values from the XAPL parameter list that are necessary to identify that unique request from the RACF access control module.
- Links SMF type 80 records with DB2 IFCID 314 records. Each invocation of the RACF access control module might produce an SMF type 80 record. DB2 might produce a DB2 IFCID 314 record in addition to the SMF type 80 records cut by RACF. You can determine that the records were cut for the same RACF access control module request if the `LOGSTR_TIME` and `LOGSTR_USER` values in the SMF type 80 record match the XAPLSTCK and XAPLUPRM values in the IFCID 314 request. The RACF access control module uses these time and user values created from the log string data to link the RACF and DB2 information.

Table 7 shows the ordered information included in log string data. A blank space separates each field, as indicated in the table.

Table 7. Information contained in log string data

Log string data	Length	XAPL field name	Description
LOGSTR_DATA	DS 0CL241		
LOGSTR_TIME	DS CL8	XAPLSTCK	Time
	DS CL1		
LOGSTR_USER	DS CL8	XAPLUPRM	User
	DS CL1		
LOGSTR_SUBSYSTEM	DS CL4	XAPLGPAT	Subsystem name, or if data sharing, DB2 group attachment name
	DS CL1		
LOGSTR_OBJTYPE	DS CL1	XAPLTYPE	Object type
	DS CL1		

Table 7. Information contained in log string data (continued)

Log string data	Length	XAPL field name	Description
LOGSTR_FLAGS	DS 0CL16	XAPLFLG1	**Flags:** The flags in this field are declared as BL1. The field is translated to CL16 in the LOGSTR data field and contains one character for each bit with a blank character between each one. • If the bit is on, Y is inserted. • If the bit is off, N is inserted. • Reserved bits are left blank.
LOGSTR_SECNDRY_ID	DS CL1		Secondary ID (Y or N)
	DS CL1		
LOGSTR_USERTAB	DS CL1		User table (Y or N)
	DS CL1		
LOGSTR_AUTOBIND	DS CL1		Autobind authority check (Y or N)
	DS CL1		
LOGSTR_DBCRTVW	DS CL1		DBADM authority to create views for others (Y or N)
	DS CL1		
LOGSTR_RDRW	DS CL1		Read/write request (Y or N)
	DS CL1		
LOGSTR_NOAUDIT	DS CL1		Suppress failure records (Y or N)
	DS CL5		
LOGSTR_OBJNAME	DS CL20	XAPLOBJN	Object name: This is the first 20 bytes of the XAPLOBJN field.
	DS CL1		
LOGSTR_OBJOWNER	DS CL20	XAPLOWNQ	Object owner or qualifier: This is the first 20 bytes of the XAPLOWNQ field.
	DS CL1		
LOGSTR_REL1	DS CL20	XAPLREL1	Related information 1: This is the first 20 bytes of the XAPLREL1 field.
	DS CL1		
LOGSTR_REL2	DS CL20	XAPLREL2	Related information 2: This is the first 20 bytes of the XAPLREL2 field.
	DS CL1		
LOGSTR_PRIV	DS CL3	XAPLPRIV	Privilege
	DS CL1		
LOGSTR_SOURCE	DS CL1	XAPLFROM	Source of the request
	DS CL1		
LOGSTR_CLASS	DS CL8		Class name
	DS CL1		
LOGSTR_ENTY	DS CL100		Entity name: This is the first resource checked for a specific request.

Examples for setting audit controls for DB2

The RACF access control module attempts to produce an audit record after checking the list of profiles.

Example 1

In this example, user ROGERM wants to use the SQL SELECT statement to retrieve information from table ICH in database DSNDB04 on the DB2 subsystem named DSN. The owner of the table is LOVES. (Refer to Appendix D, “RACF authorization checking reference” for the summary of table checking for the privilege “SELECT” on page 93.)

1. Does ROGERM own the table?

   Because ROGERM does not own the table, the table name qualifier passed from DB2 does not match the user ID. In this case, RACF does not check a profile, so no audit record is written.
2. Does ROGERM have SELECT authority?

   RACF checks DSN.LOVES.ICH.SELECT in classes MDSNTB and GDSNTB. ROGERM does not have the required SELECT authority. If ROGERM doesn’t meet any of the other requirements, this is the “first failing resource.”
3. Does ROGERM have database administrator authority?

   RACF checks DSN.DSNDB04.DBADM in class DSNADM. ROGERM does not have this authority.
4. Does ROGERM have system administrator authority?

   RACF checks DSN.SYSADM in class DSNADM. ROGERM does not have this authority.

Because ROGERM has none of the required authorities, RACF produces SMF records relating to the first failure it encountered. Although ROGERM didn’t own the table, no profiles were checked and failures were not audited. Therefore, the first failing resource is DSN.LOVES.ICH.SELECT. RACF produces an audit record for this resource and identifies it in message DSN408I. The data is contained in the log string information and can be used in a report.

Example 2

In this example, user ROGERM issues a START DATABASE(DSNDB04) request for DB2 subsystem DSN. (Refer to Appendix D, “RACF authorization checking reference” for the summary of database checking for the privilege “STARTDB” on page 74.)

1. Does ROGERM have STARTDB authority?

   RACF checks DSN.DSNDB04.STARTDB in classes MDSNDB and GDSNDB. ROGERM does not have the required STARTDB authority. If ROGERM doesn’t meet any of the other requirements, this is the “first failing resource.”
2. Does ROGERM have database maintenance authority?

   RACF checks DSN.DSNDB04.DBMAINT in class DSNADM. ROGERM does not have the required DBMAINT authority.
3. Does ROGERM have database control authority?

   RACF checks DSN.DSNDB04.DBCTRL in class DSNADM. ROGERM does not have the required DBCTRL authority.
4. Does ROGERM have database administrator authority?

   RACF checks DSN.DSNDB04.DBADM in class DSNADM. ROGERM does not have the required DBADM authority.

5. Does ROGERM have system control authority?

   RACF checks DSN.SYSCTRL in class DSNADM. ROGERM does not have this authority.
6. Does ROGERM have system administrator authority?

   RACF checks DSN.SYSADM in class DSNADM. ROGERM does not have this authority.

Because ROGERM has none of the sufficient authorities, RACF produces SMF records relating to the failure. The failure record is written for resource DSN.DSNDB04.STARTDB, which was the first failing resource. The log string information can help you to determine what ROGERM wanted to do. It includes the object type, object name, and privilege, which you can use in a report.

Chapter 10. Special considerations

In certain instances, the RACF authorization checking done by the RACF access control module is different from the authorization checking done by DB2. These instances are described in this section, along with other DB2 authorization considerations.

Matching schema names

Certain privileges associated with schema objects (such as user-defined functions, user-defined distinct types, and stored procedures), can be given if the user identity *matches* the schema name. The schema name is a short SQL identifier used as a qualifier in the name of schema objects and creates a logical grouping of these objects. It is often, but not always, a DB2 authorization ID. For applicable privileges, RACF access control module will look for a match on schema name before checking RACF profiles.

For authorization checking of the CREATEIN schema privilege, the RACF access control module first checks to see if the user identity in fields XAPLUCHK and XAPLUPRM matches the schema name in XAPLOBJN. If those two fields are equal, the RACF access control module allows the access. For all other schema privileges, the RACF access control module first checks to see if the user identity in XAPLUCHK matches the schema name in XAPLOWNQ. If those two fields are equal, the RACF access control module allows the access. In each case, when the RACF access control module allows access, it returns a return code 0 in EXPLRC1 and reason code 14 in EXPLRC2, and no further checking occurs. If the RACF access control module does not allow the access, profile checking occurs. See Appendix D, "RACF authorization checking reference," on page 67 for details.

Restriction: On multilevel-secure systems with RACF SETROPTS MLS option active, the schema match check is not performed.

Materialized query tables

When a materialized query table is created, a create view (CRTVUAUTT) authorization check is performed. The CRTVUAUTT check is used to determine whether the creator of a materialized query table can provide the required SELECT privileges on base tables to the owner of the materialized query table. If the owner of the materialized query table has the required privileges, then the CRTVUAUTT authorization check proves redundant. However, the check is performed before the owner of the materialized query table's privileges are determined. Therefore, if the materialized query table owner holds the necessary privileges and the creator of the materialized query table does not, the CRTVUAUTT check can produce unwanted error messages. To suppress the these unwanted error messages, XAPLFSUP is turned on to indicate that the RACF access control module should suppress these messages.

DB2 data sharing

The RACF access control module can be used with DB2 data sharing. When DB2 has been configured for data sharing, it will pass the RACF access control module the name of the DB2 data sharing group name in place of the DB2 subsystem name. As a result, class names and profile names must be defined with the DB2 data sharing group name in place of the DB2 subsystem name. To use the RACF access control module in this environment, all systems in the DB2 data sharing group must share the same RACF database.

For more information on DB2 data sharing, see *DB2 Data Sharing: Planning and Administration*.

PUBLIC*

The RACF access control module does not directly support use of the DB2 authorization name `PUBLIC*`, which means `PUBLIC AT ALL LOCATIONS` and is the DB2 value that represents all users in the network. However, you can define a resource profile using generic characters in multiple-subsystem scope in place of the DB2 subsystem name, with the appropriate UACC level for the object. This profile would then allow all users from all subsystems to access the resource as desired.

Authorization checking for operations on views

For most operations on views, the RACF access control module checks for authorization to the view. However, because the INSERT, DELETE, and UPDATE operations on views can affect the base tables for the views, authorization checking for these operations is different.

In general, two types of views can be defined:

- Updatable view: For example, a view that is defined with simple column references in the SELECT list of the view definition, and a single table in the FROM clause of the view definition. An INSERT, DELETE, or UPDATE operation to the view is reflected to the underlying table.
- Read-only view: For example, a view created from multiple tables. The INSERT, DELETE, and UPDATE operations fail for these views.

For INSERT, DELETE, and UPDATE operations on updatable views, the RACF access control module checks for authorization to the base table for a view, and not to the view itself. All privileges that allow a user to access the base table are checked, including:

- Ownership of the base table
- DBADM on the database that contains the base table
- SYSADM

For INSERT, DELETE, and UPDATE operations on read-only views, the RACF access control module checks for authorization to the view.

If a view is created on another view, during view creation the RACF access control module does authorization checks for INSERT, DELETE, and UPDATE. These checks are done on the base view.

For more information, see “View privileges” on page 98.

Implicit privileges of ownership

When the user is the owner of a DB2 object, the user may have some implicit privileges, but not all privileges associated with the object. The RACF access control module supports certain implicit privileges of ownership for the following DB2 objects and associated privileges.

Table 8. DB2 objects and privileges associated with implicit ownership

DB2 object	**Owner field**	**Implicit privileges**
Java archive (JAR)	XAPLREL1	USAGE
Package	XAPLREL1	BINDAUT and COPYAUT
Plan	XAPLOWNQ	BINDAUT
Sequence	XAPLREL1	ALTER, COMMENT ON, and USAGE
Stored procedure	XAPLREL1	DISPLAY, EXECUTE, START, and STOP
Table	XAPLOWNQ	All privileges except CRTSYAUT, DRPSYAUT, and CRTVUAUT
User-defined distinct type	XAPLREL1	USAGE

Table 8. DB2 objects and privileges associated with implicit ownership (continued)

DB2 object	Owner field	Implicit privileges
User-defined function	XAPLREL1	DISPLAY, EXECUTE, START, and STOP
View	XAPLOWNQ	COMMENT ON and DROP

Note: For schema objects, schema matching is similar to implicit privileges of
ownership in that if your user ID matches the schema name you can perform
certain actions on objects in the schema (such as DROP), but you cannot
use or execute the objects. For more information on schema matching, see
"Matching schema names" on page 41.

To check authorization for the privileges associated with implicit ownership, the RACF access control module first checks to see if XAPLUCHK or XAPLUPRM matches the value passed in the owner field. (See Table 8 on page 43 for the name of the field that DB2 uses to pass owner information for each object type.) If either of these two fields are equal, the RACF access control module authorizes access and returns a return code 0 in EXPLRC1 and reason code 13 in EXPLRC2. If this check fails, a check is made to see if XAPLUCHK equals the owner field ("does the current SQL ID equal the owner of the object?"). If these two fields are equal, access is allowed. If this check fails, profile checking will occur.

Restriction: On multilevel-secure systems with RACF SETROPTS MLS option active, the ownership check is not performed.

CREATETMTAB privilege

In DB2, the DBMAINT, DBCTRL, and DBADM administrative authorities are sufficient for the CREATETMTAB privilege. However, with the RACF access control module, a user must have at least *one* of the following privileges or authorities:

1. The CREATETMTAB privilege
2. The CREATETAB privilege
3. SYSCTRL authority
4. SYSADM authority

For the exact class and resource names, see Appendix D, "RACF authorization checking reference," on page 67.

CREATE VIEW privilege

If the installation option DBADM CREATE AUTH on panel DSNTIPP (ZPARM DBACRVW) is set to YES during DB2 installation, users with DBADM privilege for a database may be allowed to create views for others. See *DB2 Administration Guide* for information about other privileges required to create a view.

When a view is based on tables or a combination of tables and views from more than one database, the view creator must have DBADM for at least one database that contains a table referenced in the view.

If SYSCTRL or SYSADM authorization checking does not allow the CREATE VIEW privilege, and the XAPLCRVW field indicates that DBACRVW is enabled, the RACF access control module checks the user's DBADM authorization for each database in the list. The result of each DBADM check is placed in the XAPLDBDA field associated with each database. See Chapter 8, “Debugging the RACF access control module,” on page 31 for information about capturing the results from the RACF access control module.

CREATE ALIAS privilege

If the installation option DBADM CREATE AUTH on panel DSNTIPP (ZPARM DBACRVW) is set to YES during DB2 installation, users with DBADM or DBCTRL privilege for a database may be allowed to create aliases for others. See *DB2 Administration Guide* for information about other privileges required to create an alias.

If SYSCTRL or SYSADM authorization checking does not allow the CREATE VIEW privilege, and the XAPLCRVW field indicates that DBACRVW is enabled, the RACF access control module checks the user’s DBADM and DBCTRL authorization for the database. The result of each DBADM and DBCTRL check is placed in the XAPLDBDA field associated with each database. See Chapter 8, “Debugging the RACF access control module,” on page 31 for information about capturing the results from the RACF access control module.

"Any table" privilege

In DB2, the UPDATE or REFERENCES privilege for a specific column is sufficient to allow the “any table” privilege. When the RACF access control module is invoked for the “any table” privilege, having the UPDATE privilege or the REFERENCES privilege is not sufficient to provide the user with the “any table” privilege.

"Any schema" privilege

RACF does not perform authorization checks looking for “all privileges on all schemas” as DB2 does for the CREATEIN, ALTERIN, DROPIN, and COMMENT ON privileges on schemas; nor does RACF look for “all privileges on all stored procedures” as DB2 does for the EXECUTE privilege for stored procedures. Note that RACF generic profiles can be used to define protection for sets of similarly named schemas and stored procedures. RACF variables and RACF grouping profiles may be used for the protection attributed of schemas and stored procedures that are not similarly named.

UPDATE and REFERENCES authorization on DB2 table columns

The RACF access control module handles UPDATE and REFERENCES authorizations associated with columns by first checking for access to the entire table (example: *table*.UPDATE) and if not permitted, then to each individual column (example: *table.column*.UPDATE).

When performing an authorization check on a column privilege, the RACF access control module informs DB2 if access is allowed because it is allowed on the whole table or through an individual column. In DB2, this check is performed using fields UPDATECOLS and REFCOLS. The RACF access control module returns a value to DB2 in output field XAPLONWT.

When performing the authorization check on the entire table and authorization is given to the requestor, the RACF access control module returns a blank (‘ ’) in the output field XAPLONWT and sends a return code of 0.

If the authorization is given for a particular column or set of columns using a generic profile, the RACF access control module returns an asterisk ('*') in output field XAPLONWT and sends a return code of 0. DB2 provides the column name included in XAPLREL1 to the RACF access control module.

The XAPLDIAG output parameter

The output parameter XAPLDIAG is used to contain return codes and reason codes. When a RACROUTE REQUEST=FASTAUTH check fails to grant access, the RACF access control module records the failing SAF return code, RACF return and reason codes in XAPLDIAG. Each word of XAPLDIAG contains a FASTAUTH SAF return code (1 byte), the RACF return code (1 byte) and the RACF reason code (2 bytes), from left to right. All return codes and reason codes are shown in hexadecimal. In this way, DB2 or other programs have a way to trap and obtain diagnostic information.

See Chapter 8, "Debugging the RACF access control module," on page 31 for more information.

DB2 aliases for system-directed access

RACF applies protection to the base object, not to a DB2 alias. This is because DB2 authorization checks are made using the base object name, not the alias. By the time the RACF access control module is passed the object name, it has already been resolved from the alias name to the base name.

Considerations for remote and local resources

The RACF entity check is always performed for local resources. Remote resources are always checked by the remote DB2. This also occurs when binding an application that accesses remote resources.

DB2 GRANT statements

The RACF access control module provides RACF authorization checking of all privileges for all DB2 objects listed in "Privilege names" on page 26. However, the RACF access control module does not call RACF for DB2 **GRANT** statement checking but instead defers to DB2 authorization checking. When RACF is called by the RACF access control module, it does not use DB2 authorizations given using DB2 **GRANT** statements but uses only the resources you defined to RACF.

Structured Query Language (SQL) allows authorities to be held with the WITH GRANT option, which allows users to GRANT those privileges to others. The RACF access control module does not provide this support.

SQL supports the GRANT ALL privilege for any DB2 object. When you use the RACF access control module, you can issue a generic RACF **PERMIT** command to provide the equivalent support. The following command authorizes a user to all DB2 privileges on a DB2 table.

Example:

```
PERMIT DB2-subsystem.table-owner.table-name.* CLASS (MDSNTB) ID(userid) ACCESS(READ)
```

DB2 object names with blank characters

In DB2, it is possible to use delimited identifiers to create DB2 object names containing blank characters. However, RACF resource names cannot contain blank characters. As a result, when the RACF access control module encounters a DB2 object name containing blank characters, it will translate the blank characters to underscores (_, X'6D') before performing security checking. To protect DB2 objects containing blanks, you must define RACF profiles that will match an underscore (either explicitly or via generics) in place of the blank characters.

DB2 object names with special characters

In DB2, it is possible to use delimited identifiers to create DB2 object names containing any character in the EBCDIC syntactic character set. However, RACF does not allow the use of commas, semicolons, or parentheses in resource names. To protect DB2 objects containing these characters (or any other characters that are not allowed by the RACF command processors), you need to define RACF profiles containing generic characters that will match the unsupported characters.

Authority checking for all packages in a collection

The naming convention for DB2 package objects is:

```
subsystem-name.collection-ID.package-ID.privilege-name
```

When a DB2 user tries to perform an operation on all packages in a collection, DB2 may pass an asterisk (*****) to the RACF access control module in place of *package-ID*. To ensure consistent results between the RACF access control module and the RACF command processors (SEARCH and RLIST), the asterisk (*****) in the resource name should match the asterisk (*****) in the profile name.

For example, in DB2, you can BIND a plan using all of the packages from a given collection. When that plan is subsequently executed, DB2 will check the user's authority to execute all packages in the collection by passing an asterisk (*****) in place of the collection name. For example, suppose the following DB2 commands are issued for subsystem DSN:

```
BIND PACKAGE(DSNTEP2) MEMBER(DSNTEP2) ACT(REP) ISO(CS)
BIND PLAN(DSNTEP42) PKLIST(DSNTEP2.*) ACT(REP) ISO(CS)
RUN  PROGRAM(DSNTEP2) PLAN(DSNTEP42) -
```

When DB2 gets to the execution step, it invokes the RACF access control module to check the user's authority to EXECUTE package `DSNTEP2.*`, where the asterisk (*****) means all packages in the collection.

The RACF access control module checks the user's authority to resource:

```
DSN.DSNTEP2.*.EXECUTE     (in class MDSNPK)
```

The RACF profile name protecting this resource should contain a single asterisk (*****) to match the asterisk (*****) in the resource name.

AUTOBIND requests for user-defined functions

RACF fails all authorization checks associated with AUTOBIND requests for user-defined functions. That is, when:

- XAPLAUTO (in XAPLFLG1) is non-zero,
- XAPLTYPE indicates a function ("F"), and

- XAPLPRIV is 64 (EXECUTE)

then a return code 8 and reason code 17 are returned, and no resource check is performed. This causes the AUTOBIND request to fail. A manual REBIND is then required.

Identity used for authorization checks

The RACF access control module receives user identification information in the XAPL (DSNXAPL) parameter list that is passed by DB2. In the XAPL, the RACF access control module receives:

- a pointer to the input ACEE that represents the identity of the requester (XAPLUPRM).
- the 1–8-character user ID of the requester (XAPLUPRM).

  Note: The XAPLUPRM value is used for all RACF authorization checking, although RACF actually checks the input ACEE itself to determine this identity. The identity represented by the ACEE is the same as the user ID passed in XAPLUPRM.
- the 1–128-character authorization ID (XAPLUCHK) that DB2 uses for the authorization check. The XAPLUCHK may contain a value that is not a RACF user ID or group, and it may differ from the XAPLUPRM.

While the RACF access control module uses the XAPLUCHK and XAPLUPRM values to perform ownership checks, it performs all access authorization checks using only XAPLUPRM.

It is possible for the XAPLUCHK value to be different from the user ID (XAPLUPRM) represented in the ACEE pointed to by XAPLACEE. For example, this can occur when a BIND request is issued and the binder is not the owner of the plan or package. The RACF access control module is invoked to determine whether the binder is authorized to do the BIND. If this check is successful, it is then invoked to check the binder's authorization to access each DB2 resource accessed in the plan or package. For the BIND check, XAPLUPRM and XAPLUCHK have the authorization ID of the binder. However, for the subsequent checks on the DB2 resources accessed in the plan or package, XAPLUPRM still has the authorization ID of the binder, but XAPLUCHK now has the authorization ID of the plan or package owner. For the BIND to succeed, the binder must have authorization to bind this plan or package, and be authorized to access all DB2 resources accessed in it. DB2 authorization performs the subsequent checks on the owner of the plan/package and not the binder.

When DB2 cannot provide an ACEE

Sometimes DB2 cannot provide an ACEE. For example, if you are not using external security in CICS® (for example, `SEC=NO` is specified in the DFHSIT), CICS does not pass an ACEE to the CICS attachment facility. When DB2 does not have an ACEE, it passes zeros in the XAPLACEE field. If this happens, your routine can return a 4 in the EXPLRC1 field, and let DB2 handle the authorization check.

Restrictions:

1. The ACEE address is not passed for IMS™ transactions.
2. The ACEE address is passed for CICS transactions, when available. If you implement the DB2 CICS attachment facility and CICS is configured to use an external security manager, such as RACF, DB2 passes the ACEE address, if available.

3. The ACEE address is passed for DB2 commands, when available. If the master console is used, DB2 does not pass the ACEE address because an ACEE is not available. However, if the user signs on to an MVS operator console, DB2 passes the ACEE address, if available.

Authorization ID, ACEE relationship

XAPL has two authorization ID fields, XAPLUPRM (the primary authorization ID) and XAPLUCHK (the authorization ID that DB2 uses to perform the authorization). These two fields might have different values.

The ACEE passed in XAPLACEE is that of the primary authorization ID, XAPLUPRM.

Invalid or inoperative plans and packages

In DB2, when a privilege required by a plan or package is revoked, the plan or package is invalidated. If you use an authorization access control routine, it cannot tell DB2 that a privilege is revoked. Therefore, DB2 cannot know to invalidate the plan or package.

If the revoked privilege was EXECUTE on a user-defined function, DB2 marks the plan or package inoperative instead of invalid.

If a privilege that the plan or package depends on is revoked, and if you want to invalidate the plan or package or make it inoperative, you must use the SQL GRANT statement to grant the revoked privilege and then use the SQL REVOKE statement to revoke it.

Dropping views

In DB2, when a privilege required to create a view is revoked the view is dropped. Similar to the revocation of plan privileges, such an event is not communicated to DB2 by the authorization checking routine.

If you want DB2 to drop the view when a privilege is revoked, you must use the SQL statements GRANT and REVOKE.

Caching of EXECUTE on plans

The results of authorization checks on the EXECUTE privilege are not cached when those checks are performed by the exit routine.

Caching of EXECUTE on packages and routines

The results of authorization checks on the EXECUTE privilege for packages and routines are cached (assuming that package and routine authorization caching is enabled on your system). If this privilege is revoked in the exit routine, the cached information is not updated to reflect the revoke. You must use the SQL GRANT and REVOKE statements to update the cached information.

Caching of dynamic SQL statements

Dynamic statements can be cached when they have passed the authorization checks (assuming that dynamic statement caching is enabled on your system). If the privileges that this statement requires are revoked from the authorization ID that is cached with the statement, then this cached statement must be invalidated. If the privilege is revoked in the exit routine this does not happen, and you must use the SQL statements GRANT and REVOKE to refresh the cache.

Resolution of user-defined functions

The create timestamp for the user-defined function must be older than the bind timestamp for the package or plan in which the user-defined function is invoked. If DB2 authorization checking is in effect, and DB2 performs an automatic rebind on a plan or package that invokes a user-defined function, any user-defined functions that were created after the original BIND or REBIND of the invoking plan or package are not candidates for execution.

If you use an access control authorization exit routine, some user-defined functions that were not candidates for execution before the original BIND or REBIND of the invoking plan or package might become candidates for execution during the automatic rebind of the invoking plan or package. If a user-defined function is invoked during an automatic rebind, and that user-defined function is invoked from a trigger body and receives a transition table, the form of the invoked function that DB2 uses for function selection includes only the columns of the transition table that existed at the time of the original BIND or REBIND of the package or plan for the invoking program.

Initialization

To indicate the function to be performed, DB2 passes one of three function codes to the RACF access control module—for initialization, authorization checking, or termination. For general information about initialization and termination information, see Chapter 1, “Overview,” on page 1.

Any DB2 classes you want to use must be active during RACF access control module initialization (`XAPLFUNC=1`). You cannot activate a DB2 class later and expect the RACF access control module to perform authorization checking against it, because the class will not be RACLISTed. RACLISTing is only done during initialization of the RACF access control module.

To start using DB2 classes that were not previously RACLISTed during initialization, you will need to stop and restart DB2.

Once the DB2 subsystem has initialized, the following command needs to be issued to affect profile changes for classes being used by the RACF access control module:

```
SETROPTS RACLIST(classname) REFRESH
```

The following informational messages are issued for each initialization: IRR908I, IRR909I, IRR910I, and IRR911I.

Note: The classes listed in message IRR911I may be a valid subset of the classes listed in message IRR910I. The RACF access control module is programmed to RACLIST all supported DB2 classes. Message IRR910I lists the DB2 classes for which the RACF access control module has initiated RACLIST.

However, message IRR911I lists only the DB2 classes that were successfully RACLISTed. In order to be successfully RACLISTed, a DB2 class must be active and contain at least one profile. Therefore, there are valid circumstances where the list of classes contained in IRR911I will be a subset of those listed in IRR910I.

Failure to initialize

If the RACF access control module fails to initialize for any reason, messages IRR900A, IRR901A, IRR902A, and IRR903A are issued to the security console. If this occurs, you can do the following:

1. Check that the DB2 classes are active, and that there is at least one profile defined in each class.
2. Examine RACROUTE REQUEST=LIST return and reason codes to determine why RACLISTing of classes is failing in the RACF access control module.
3. Check if any other required resources (GETMAIN, for example) are obtainable.

Return codes and reason codes from initialization

Return codes from the RACF access control module are returned in the DB2-supplied EXPL field EXPLRC1. Reason codes from the RACF access control module are returned in the DB2-supplied EXPL field EXPLRC2. See Appendix A, “XAPLFUNC reference,” on page 53 for the meanings of the return and reason codes from the initialization of the RACF access control module.

Deferring to native DB2 authorization

Deferring to native DB2 authorization may or may not require removal of the RACF access control module. A return code of 4 from the RACF access control module indicates that DB2 should defer to DB2 security checking for that particular authorization check.

Removing the RACF access control module

If the RACF access control module is removed, DB2 reverts to using native DB2 authorization, in which authority is determined by DB2 catalogs.

In addition, you may need to:

1. Inactivate any classes related to the DB2 processing
2. Make the necessary GRANTs in DB2

Common problems and considerations

Common problems that could occur as a result of defining special classes in the class descriptor table (CDT) follow:

- A class is not defined in the CDT.

  This results in a return code of 4 (profile not found) from the RACF access control module.
- If class is defined in the static CDT, there is incorrect linkage editor procedures from the CDT.
- If class is defined in the static CDT, it is link-edited properly but a re-IPL has not occurred to pick up the changes.
- If class is defined in the dynamic CDT, the CDTINFO class was not RACLISTed or refreshed to pick up the changes.

- Single-subsystem scope class names are being used and a new subsystem is using the RACF access control module before classes for the subsystem have been defined.
- Messages IRR900A, IRR901A, IRR902A, and IRR903A are issued because the RACF access control module could not initialize correctly.
  1. Check to see if DB2 classes are active.
  2. Determine if and why RACLISTing of classes is failing in the module by examining RACROUTE REQUEST=LIST return and reason codes.
  3. Check to see if any other required resources (such as GETMAIN, for example) are obtainable.

Appendix A. XAPLFUNC reference

DB2 calls the RACF access control module using the following function codes. Table 9 shows the purpose and timing of each function call.

Table 9. XAPLFUNC codes and corresponding functions

Function code	Time of call	Purpose
`XAPLFUNC=1`	DB2 initialization	Create in-storage profiles and indicate what action DB2 should take if the RACF access control module abends or fails to initialize.
`XAPLFUNC=2`	DB2 authorization	Check DB2 objects and authorities.
`XAPLFUNC=3`	DB2 termination	Delete in-storage profiles.

Unsupported function codes: If the RACF access control module receives a XAPLFUNC function code other than 1, 2 or 3, the RACF access control module sends a return code of 12 to the caller.

When a return code of 12 is received:

- Native DB2 authorization is used if `&ERROROPT 1` or the level of DB2 is below DB2 Version 7.
- The DB2 subsystem stops if `&ERROROPT 2` and the level of DB2 is DB2 Version 7 or later.

Initialization (XAPLFUNC = 1)

When the RACF access control module is called with XAPLFUNC function code of 1, it issues a RACROUTE REQUEST=STAT request to determine if RACF is active. If RACF is not active, the RACF access control module returns to DB2 with a return code of 12. If RACF is active, the RACF access control module builds the class names, as specified by the assembler SET symbols, and performs a RACROUTE REQUEST=LIST,CLASS=*classname* for each new DB2-related class.

Attention

- If you override `&CLASSNMT` or use the single-subsystem scope, the RACF access control module uses only installation-defined classes.
- If you use the multiple-subsystem scope with the default `&CLASSNMT`, the RACF access control module uses classes supplied by IBM.

See *z/OS Security Server RACF Security Administrator's Guide* for a list of DB2 classes supplied by IBM.

The RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES request brings profiles to a data space for that particular DB2 or allows a subsequent DB2 to use those in-storage profiles.

If no DB2-related classes were active, a failure occurs and the RACF access control module ends with a return code of 12.

Note: The following are not failures:

- A class is not active (SAF RC=4, RACF RC=10)

- A class is not defined (SAF RC=4, RACF RC=8)

  If a class is not active or does not exist for an object or authority, the RACF access control module defers to DB2 for authorization checking and ends with a return code of 4.

If *one* request fails, the *entire* initialization fails. When this happens, the RACF access control module cleans up all the resources and ends with a return code of 12.

If you want to use DB2 classes for authorization against DB2 objects, the classes must be active when the subsystem is started.

Failures during initialization processing are indicated by a return and reason code pair and a message.

Initialization return and reason codes

The following return and reason codes are shown in decimal notation.

Return code	Meaning
0	Initialization successful.

Reason code	Meaning
0	Installation option `&ERROROPT` was set to `1`. Therefore, native DB2 authorization is used in the event of an error.
16	Installation option `&ERROROPT` was set to `2`. Therefore, the DB2 system is requested to stop in the event of an error on a subsequent authorization check.

Return code	Meaning
12	Initialization unsuccessful; don't call RACF access control module again.

Reason code	Meaning
1	An input DB2 subsystem ACEE was not provided. Installation option `&ERROROPT` was set to `1`. Therefore, native DB2 authorization will be used.
2	RACF is not active. Installation option `&ERROROPT` was set to `1`. Therefore, native DB2 authorization will be used.
3	RACROUTE REQUEST=LIST,ENVIR=CREATE failure. Installation option `&ERROROPT` was set to `1`. Therefore, native DB2 authorization will be used.
4	No active DB2 classes. Installation option `&ERROROPT` was set to `1`. Therefore, native DB2 authorization will be used.
10	Incorrect XAPL level. The value of XAPLLVL is less than `V8R1M0`. Installation option `&ERROROPT` was set to `1`. Therefore, native DB2 authorization will be used.
12	Input DB2 subsystem ACEE was not valid. Installation option `&ERROROPT` was set to `1`.

	Therefore, native DB2 authorization will be used. DB2 authorization will be used.
16	An initialization error occurred. Installation option `&ERROROPT` was set to `2`. Therefore, the DB2 subsystem is requested to stop.

Authorization checking (XAPLFUNC = 2)

The RACF access control module requires an input ACEE to perform authority checking. When an input ACEE (XAPLACEE) is not provided to the RACF access control module, it defers to DB2 for authority checking (EXPLRC1 set to 4). See *DB2 Administration Guide* for the requests for which the input ACEE (XAPLACEE) is set to zero. For these requests, authority checking must be implemented using the DB2 GRANT and REVOKE statements. RACF profiles defined for these requests will *not* be used.

The RACF access control module performs FASTAUTH checks during authorization according to the rules described in Appendix D, "RACF authorization checking reference," on page 67. In DB2, there is no concept of negative access level. RACF access control module processing ends when FASTAUTH returns a return code of 0 or the list of checks for the request has been exhausted. Failure audit records are only created for the first failing resource. All audit records associated with the same invocation of the RACF access control module contain the same LOGSTR data. See Appendix C, "Authorization processing examples," on page 61 for examples.

Authorization return and reason codes

The following return and reason codes are shown in decimal notation.

Return code	**Meaning**
0	Access permitted

Reason code	**Meaning**
0	Access permitted by FASTAUTH checking.
13	Access permitted by implicit privilege of ownership.
14	Access permitted because current SQL ID matches schema name.

4	Unable to determine; perform DB2 authorization checking

Reason code	**Meaning**
0	Input class (XAPLTYPE) not active.
11	Input ACEE (XAPLACEE) not provided.
14	The ALET could not be created for cross memory ACEE.
15	Input privilege code (XAPLPRIV) or input class (XAPLTYPE) not defined to the RACF access control module.
16	Input privilege code (XAPLPRIV) does not contain any rules.

8	Access denied

Reason code **Meaning**

0	Access denied.
17	Autobind indicator (XAPLAUTO) is not zero, indicating AUTOBIND was requested. Manual REBIND is required.

FASTAUTH return code translation

Each time the RACF access control module is invoked, it may in turn invoke RACROUTE REQUEST=FASTAUTH multiple times. If one of the FASTAUTH requests completes with a return code of zero, the return code passed back to DB2 will be zero. If none of the FASTAUTH requests complete with a return code of zero, the collection of return codes from FASTAUTH must be translated into a single resultant return code. Return code translation can be summarized as follows:

If all object resource checks result in a return code of 4 and none of the DSNADM checks result in a return code of 0, the RACF access control module passes back a return code of 4.

If at least one object resource check results in a return code of 8 and none of the DSNADM checks result in a return code of 0, the RACF access control module passes back a return code of 8.

If no object resource profiles are checked and all of the DSNADM checks result in a return code of 8, the RACF access control module will pass back a return code of 8. Otherwise, if no object resources are checked and the DSNADM checks result in a mix of 4s and 8s, the RACF access control module passes back a return code of 4.

All failing SAF/RACF return codes and RACF reason codes are placed in the output parameter field in XAPLDIAG, to be returned to DB2. This information is then available to DB2, SQL, or other programs to obtain diagnostic information from it.

Table 10 illustrates the method used to do this translation.

Table 10. FASTAUTH return code translation

Return code from object profile	Return code from ADM profile	Output return code
—	All 4s	04
—	All 8s	08
—	Mix	04
All 4s	All 4s	04
All 4s	All 8s	04
All 4s	Mix	04
All 8s	All 4s	08
All 8s	All 8s	08
All 8s	Mix	08
Mix	All 4s	08
Mix	All 8s	08
Mix	Mix	08

Note: *Mix* indicates a variety of 4 and 8 return codes.

Termination (XAPLFUNC = 3)

When the RACF access control module module uses XAPLFUNC function code 3, it issues a RACROUTE REQUEST=LIST,ENVIR=DELETE,GLOBAL=YES request. The classes that were previously brought into storage during DB2 initialization are deleted.

Failures during termination processing are indicated by a return and reason code pair and a message.

Termination return and reason codes

The following return and reason codes are shown in decimal notation.

Return code	Meaning
0	Termination successful
8	Termination failure

Reason code	Meaning
1	Input DB2 subsystem ACEE was not provided.
7	RACROUTE REQUEST=LIST,ENVIR=DELETE failure.
12	Input DB2 subsystem ACEE was not valid.

Appendix B. Supplied RACF resource classes for DB2

The following RACF classes for DB2 objects and administrative authorities are supplied in the class descriptor table (CDT).

Table 11. Resource classes for DB2 objects and administrative authorities

Class name	Description
DSNADM	DB2 administrative authority class
DSNR	Controls access to DB2 subsystems
GDSNBP	Grouping class for DB2 buffer pool privileges
GDSNCL	Grouping class for DB2 collection privileges
GDSNDB	Grouping class for DB2 database privileges
GDSNJR	Grouping class for Java archive files (JARs)
GDSNPK	Grouping class for DB2 package privileges
GDSNPN	Grouping class for DB2 plan privileges
GDSNSC	Grouping class for DB2 schemas privileges
GDSNSG	Grouping class for DB2 storage group privileges
GDSNSM	Grouping class for DB2 system privileges
GDSNSP	Grouping class for DB2 stored procedure privileges
GDSNSQ	Grouping class for DB2 sequences
GDSNTB	Grouping class for DB2 table, index, or view privileges
GDSNTS	Grouping class for DB2 tablespace privileges
GDSNUF	Grouping class for DB2 user-defined function privileges
GDSNUT	Grouping class for DB2 user-defined distinct type privileges
MDSNBP	Member class for DB2 buffer pool privileges
MDSNCL	Member class for DB2 collection privileges
MDSNDB	Member class for DB2 database privileges
MDSNJR	Member class for Java archive files (JARs)
MDSNPK	Member class for DB2 package privileges
MDSNPN	Member class for DB2 plan privileges
MDSNSC	Member class for DB2 schema privileges
MDSNSG	Member class for DB2 storage group privileges
MDSNSM	Member class for DB2 system privileges
MDSNSP	Member class for DB2 stored procedure privileges
MDSNSQ	Member class for DB2 sequences
MDSNTB	Member class for DB2 table, index, or view privileges
MDSNTS	Member class for DB2 tablespace privileges
MDSNUF	Member class for DB2 user-defined function privileges
MDSNUT	Member class for DB2 user-defined distinct type privileges

Appendix C. Authorization processing examples

- Examples 1 through 4 show authority checks performed on tables using supplied classes for multiple-subsystem scope (`&CLASSOPT 2`).
- Example 5 shows authority checks performed on tables using installation-defined classes for multiple-subsystem scope (`&CLASSOPT 2`).
- Example 6 shows authority checks performed on tables using installation-defined classes for single-subsystem scope (`&CLASSOPT 1`).

Example 1: Allowing access (auditing for failures)

This example shows how the RACF access control module allows access to a DB2 object (a table) based on a DB2 administrative authority profile. Auditing is activated for failures.

In this example, user ID MIKEJ is trying to alter a table called BDA0828.EMP in database JBW2000.

Setup

- Classification model (`&CLASSOPT`): `2`
- Class name root (`&CLASSNMT`): `DSN`
- Class name suffix (`&CHAROPT`): `1`

  This is the default value, but it is not used with supplied classes.
- DB2 subsystem name: `VHH1`
- **Profiles:**
  - Defined in the MDSNTB class:

    ```
    VHH1.BDA0828.EMP.ALTER
    - AUDIT(FAILURES(READ))
    - UACC(NONE)
    ```
  - Defined in the DSNADM class:

    ```
    VHH1.SYSADM
    - AUDIT(FAILURES(READ))
    - UACC(NONE)
    - ID(MIKEJ) ACCESS(READ)
    ```
- User ID MIKEJ has SYSADM authority.

Profile checking

RACF checks the following resources:

1. VHH1.BDA0828.EMP.ALTER in class MDSNTB

   Results:
   - Access is denied (return code 8).
   - No failure message (ICH408I) is issued.
   - No audit records are created.
2. VHH1.JBW2000.DBADM in class DSNADM

   Results:
   - No profile is found (return code 4).
   - No failure message (ICH408I) is issued.
   - No audit records are created.
3. VHH1.SYSADM in class DSNADM

   Results:

- Access is granted (return code 0).
- No failure message (ICH408I) is issued.
- No audit records are created.

Final result

The RACF access control module sends a return code of 0 to DB2.

Example 2: Allowing access (auditing for all attempts)

This example shows how the RACF access control module allows access to a DB2 object (a table) based on a DB2 administrative authority profile. Auditing is activated for all access attempts.

In this example, user ID MIKEJ is trying to alter a table called BDA0828.EMP in database JBW2000.

Setup

- Classification model (`&CLASSOPT`): `2`
- Class name root (`&CLASSNMT`): `DSN`
- Class name suffix (`&CHAROPT`): `1`

  This is the default value, but it is not used with supplied classes.
- DB2 subsystem name: `VHH1`
- **Profiles:**
  - Defined in the MDSNTB class:
    ```
    VHH1.BDA0828.EMP.ALTER
    - AUDIT(ALL(READ))
    - UACC(NONE)
    - ID(MIKEJ) ACCESS(NONE)
    ```
  - Defined in the DSNADM class:
    ```
    VHH1.SYSADM
    - AUDIT(ALL(READ))
    - UACC(NONE)
    - ID(MIKEJ) ACCESS(READ)
    ```
- User ID MIKEJ has SYSADM authority.

Profile checking

RACF checks the following resources:

1. VHH1.BDA0828.EMP.ALTER in class MDSNTB

   Results:
   - Access is denied (return code 8).
   - No failure message (ICH408I) is issued.
   - No audit records are created.
2. VHH1.JBW2000.DBADM in class DSNADM

   Results:
   - No profile is found (return code 4).
   - No failure message (ICH408I) is issued.
   - No audit records are created.
3. VHH1.SYSADM in class DSNADM

   Results:
   - Access is granted (return code 0).

- No failure message (ICH408I) is issued.
- An audit record is created, which includes the following log string data:
  - The VHH1.BDA0828.EMP.ALTER profile name
  - Input parameters identifying the request from DB2.

Final result

The RACF access control module sends a return code of 0 to DB2.

Example 3: Denying access

This example shows how the RACF access control module denies access to a DB2 object (a table). Auditing is activated for all access attempts.

In this example, user ID MIKEJ is trying to alter a table called BDA0828.EMP in database JBW2000.

Setup

- Classification model (`&CLASSOPT`): `2`
- Class name root (`&CLASSNMT`): `DSN`
- Class name suffix (`&CHAROPT`): `1`

  This is the default value, but it is not used with supplied classes.
- DB2 subsystem name: `VHH1`
- Profile:
  - Defined in the MDSNTB class:

    ```
    VHH1.BDA0828.EMP.ALTER
    - AUDIT(ALL(READ))
    - UACC(NONE)
    - ID(MIKEJ) ACCESS(NONE)
    ```
- User ID MIKEJ has SYSADM authority.

Profile checking

RACF checks the following resources:

1. VHH1.BDA0828.EMP.ALTER in class MDSNTB

   Results:
   - Access is denied (return code 8).
   - No failure message (ICH408I) is issued.
   - No audit records are created.
2. VHH1.JBW2000.DBADM in class DSNADM

   Results:
   - No profile is found (return code 4).
   - No failure message (ICH408I) is issued.
   - No audit records are created.
3. VHH1.SYSADM in class DSNADM

   Results:
   - No profile is found (return code 4).
   - No failure message (ICH408I) is issued.
   - No audit records are created.
4. VHH1.BDA0828.EMP.ALTER in class MDSNTB

   Results:
   - Access is denied (return code 8).

- Failure message (ICH408I) is issued.
- An audit record is created, which includes the following log string data:
  - The VHH1.BDA0828.EMP.ALTER profile name
  - Input parameters identifying the request from DB2.

Final result

The RACF access control module sends a return code of 8 to DB2.

Example 4: Deferring to DB2

This example shows how the RACF access control module defers to native DB2 authorization checking because the DB2 object (a table) is not protected by RACF.

In this example, user ID MIKEJ is trying to alter a table called BDA0828.EMP in database JBW2000.

Setup

- Classification model (`&CLASSOPT`): `2`
- Class name root (`&CLASSNMT`): `DSN`
- Class name suffix (`&CHAROPT`): `1`

  This is the default value, but it is not used with supplied classes.
- DB2 subsystem name: `VHH1`
- **Profiles:**
  - Defined in the MDSNTB class:

    `VHH1.BDA0828.EMP.ALTER`
  - Defined in the DSNADM class:

    `VHH1.SYSADM`

    `- AUDIT(ALL(READ))`
- User ID MIKEJ has SYSADM authority.

Profile checking

RACF checks the following resources:

1. VHH1.BDA0828.EMP.ALTER in class MDSNTB

   Results:
   - No profile is found (return code 4).
   - No failure message (ICH408I) is issued.
   - No audit records are created.
2. VHH1.JBW2000.DBADM in class DSNADM

   Results:
   - No profile is found (return code 4).
   - No failure message (ICH408I) is issued.
   - No audit records are created.
3. VHH1.SYSADM in class DSNADM

   Results:
   - No profile is found (return code 4).
   - No failure message (ICH408I) is issued.
   - No audit records are created.

Final result

The RACF access control module sends a return code of 4 to DB2.

Example 5: Allowing access (multiple-subsystem scope)

This example shows how the RACF access control module allows access to a DB2 object (a table) based on a DB2 administrative authority profile. The installation has defined classes MSLH1TB1 and SLH1ADM1. Auditing is activated for all access attempts.

In this example, user ID MIKEJ is trying to alter a table called BDA0828.EMP in database JBW2000.

Setup

- Classification model (`&CLASSOPT`): `2`
- Class name root (`&CLASSNMT`): `SLH1`
- Class name suffix (`&CHAROPT`): `1`
- DB2 subsystem name: `VHH1`
- **Profiles:**
  – Defined in the MSLH1TB1 class:
    ```
    VHH1.BDA0828.EMP.ALTER
    - AUDIT(ALL(READ))
    - UACC(NONE)
    ```
  – Defined in the SLH1ADM1 class:
    ```
    VHH1.SYSADM
    - AUDIT(ALL(READ))
    - UACC(NONE)
    - ID(MIKEJ) ACCESS(READ)
    ```
- User ID MIKEJ has SYSADM authority.

Profile checking

RACF checks the following resources:

1. VHH1.BDA0828.EMP.ALTER in class MSLH1TB1

   Results:
   - Access is denied (return code 8).
   - No failure message (ICH408I) is issued.
   - No audit records are created.
2. VHH1.JBW2000.DBADM in class SLH1ADM1

   Results:
   - No profile is found (return code 4).
   - No failure message (ICH408I) is issued.
   - No audit records are created.
3. VHH1.SYSADM in class SLH1ADM1

   Results:
   - Access is granted (return code 0).
   - No failure message (ICH408I) is issued.
   - An audit record is created, which includes the following log string data:
     – The VHH1.BDA0828.EMP.ALTER profile name
     – Input parameters identifying the request from DB2.

Final result

The RACF access control module sends a return code of 0 to DB2.

Example 6: Allowing access (single-subsystem scope)

This example shows how the RACF access control module allows access to a DB2 object (a table) based on a DB2 administrative authority profile. The installation has defined classes MVHH1TB1 and VHH1ADM1. Auditing is activated for all access attempts.

In this example, user ID MIKEJ is trying to alter a table called BDA0828.EMP in database JBW2000.

Setup

- Classification model (`&CLASSOPT`): `1`
- Class name root (`&CLASSNMT`): `DSN`

  This is the default value, but it is not used in single-subsystem scope.
- Class name suffix (`&CHAROPT`): `1`
- DB2 subsystem name: `VHH1`
- **Profiles:**
  - Defined in the MVHH1TB1 class:

    ```
    VHH1.BDA0828.EMP.ALTER
    - AUDIT(ALL(READ))
    - UACC(NONE)
    ```
  - Defined in the VHH1ADM1 class:

    ```
    SYSADM
    - AUDIT(ALL(READ))
    - UACC(NONE)
    - ID(MIKEJ) ACCESS(READ)
    ```
- User ID MIKEJ has SYSADM authority.

Profile checking

RACF checks the following resources:

1. BDA0828.EMP.ALTER in class MVHH1TB1

   Results:
   - Access is denied (return code 8).
   - No failure message (ICH408I) is issued.
   - No audit records are created.
2. JBW2000.DBADM in class VHH1ADM1

   Results:
   - No profile is found (return code 4).
   - No failure message (ICH408I) is issued.
   - No audit records are created.
3. SYSADM in class VHH1ADM1

   Results:
   - Access is granted (return code 0).
   - No failure message (ICH408I) is issued.
   - An audit record is created, which includes the following log string data:
     - The VHH1.BDA0828.EMP.ALTER profile name
     - Input parameters identifying the request from DB2.

Final result

The RACF access control module sends a return code of 0 to DB2.

Appendix D. RACF authorization checking reference

How to use this reference

This appendix includes information about the RACF authorization checking through the RACF access control module for the following DB2 objects:

B	Buffer pools
C	Collections
D	Databases
E	User-defined distinct types
F	User-defined functions
J	Java archives (JARs)
K	Packages
M	Schemas
O	Stored procedures
P	Application plans
Q	Sequences
R	Tablespaces
S	Storage groups
T	Tables
U	Systems
V	Views

The sections that follow outline the series of authorization checks that occur in the RACF access control module to determine if the requesting user is authorized to use a particular DB2 privilege against a particular DB2 object type. If any authorization check in the series is successful, the privilege is granted. For examples of authorization processing in the RACF access control module, see Appendix C, “Authorization processing examples,” on page 61.

In order to perform authorization checks, the RACF access control module uses the values passed with the following parameters to determine the DB2 object types and privileges:

XAPLTYPE	DB2 object type
XAPLPRIV	DB2 privilege

Restriction: The sections that follow show only the *name* of each DB2 privilege passed with the XAPLPRIV parameter. The RACF access control module uses a numeric XAPLPRIV value. See the DB2 macro DSNXAPRV in *prefix*.SDSNMACS to find the numeric value associated with each DB2 privilege name.

The profile name formats shown in this appendix are applicable if you are using multiple-subsystem scope (`&CLASSOPT 2`). If you are using single-subsystem scope (`&CLASSOPT 1`), the resource name does not include the DB2 subsystem name. If you are using DB2 data sharing, substitute *DB2-group-attachment-name* for *DB2-subsystem* in the profile name formats shown in this appendix.

How to set the level of access

When the system is configured with the RACF MLS option not active, access to
DB2 objects, privileges or administrative authorities is allowed if the user or group
requesting access is in the access list of the RACF profile protecting the object,
privilege or authority with at least READ access. If the system is configured with the
RACF MLS option active, any operation that performs a write operation (such as
UPDATE to a table) must have UPDATE authority (rather than READ).

Note: Use of UPDATE access regardless of the configuration rather than READ in
one configuration and UPDATE in another has no effect on access protection
and eases administration.

#

Buffer pool privileges

Resources: Buffer pools

Resource type: B

DB2 privileges

USE

XAPLPRIV value: **USEAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.buffer-pool-name.USE	MDSNBP or GDSNBP
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Collection privileges

Resources: Collections

Resource type: C

DB2 administrative authorities

PACKADM

XAPLPRIV value: **PKADMAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSADM	DSNADM

DB2 privileges

CREATE IN

XAPLPRIV value: **CRTINAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.collection-ID.CREATEIN	MDSNCL or GDSNCL
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Database privileges

Resources: Databases

Resource type: D

DB2 administrative authority

DBCTRL

XAPLPRIV value: **DBCTLAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DB2 privileges

CREATETAB

XAPLPRIV value: **CRTTBAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.CREATETAB	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CHANGE NAME QUALIFIER

XAPLPRIV value: **QUALAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CREATETS

XAPLPRIV value: **CRTTSAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.CREATETS	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DISPLAYDB

XAPLPRIV value: **DSPDBAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.DISPLAYDB	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM

One of these resources:	In class:
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.DISPLAY	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DROP

XAPLPRIV value: **DROPAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.DROP	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

IMAGCOPY, MERGECOPY, MODIFY RECOVERY, QUIESCE

XAPLPRIV values: **IMCOPAUT, MERGEAUT, MODAUT, QUIESAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.IMAGCOPY	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

RECOVERDB, REPORT

XAPLPRIV values: **RECDBAUT, REPRTAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.RECOVERDB	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

REORG

XAPLPRIV value: **REORGAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.REORG	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

REPAIR, RUN REPAIR UTILITY
XAPLPRIV values: **REPARAUT, DIAGAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.REPAIR	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

REPAIR DBD
XAPLPRIV value: **RDBDAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

RUN CHECK UTILITY, STATS
XAPLPRIV values: **CHECKAUT, STATSAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.STATS	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

STARTDB
XAPLPRIV value: **STARTAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.STARTDB	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

STOPDB

XAPLPRIV value: **STOPAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.STOPDB	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

TERM UTILITY

XAPLPRIV value: **TERMAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

TERM UTILITY ON DATABASE

XAPLPRIV value: **TERMDAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM

Java archive (JAR) privileges

Resources: Java archives (JARs)

Resource type: J

DB2 privileges

USAGE

XAPLPRIV value: **USAGEAUT**

Does the user own the Java archive (JAR)?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.schema-name.JAR-name.USAGE	MDSNJR or GDSNJR
DB2-subsystem.SYSADM	DSNADM

Package privileges

Resources: Packages

Resource type: K

DB2 privileges

BIND

XAPLPRIV value: **BINDAUT**

Does the user own the package?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.collection-ID.package-ID.BIND	MDSNPK or GDSNPK
DB2-subsystem.owner.BINDAGENT	MDSNSM or GDSNSM
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

COPY

XAPLPRIV value: **COPYAUT**

Does the user own the package?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.collection-ID.package-ID.COPY	MDSNPK or GDSNPK
DB2-subsystem.owner.BINDAGENT	MDSNSM or GDSNSM
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DROP

XAPLPRIV value: **DROPAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

EXECUTE

XAPLPRIV value: **CHKEXEC**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.collection-ID.package-ID.EXECUTE	MDSNPK or GDSNPK
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSADM	DSNADM

All package privileges (PACKADM or SYSADM)

XAPLPRIV value: **ALLPKAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.collection-ID.PACKADM	DSNADM
DB2-subsystem.SYSADM	DSNADM

All package privileges (PACKADM, SYSADM, or SYSCTRL)

XAPLPRIV value: **SUBPKAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.collection-ID.PACKADM The user has authority to *collection-ID*.	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Plan privileges

Resources: Application plans

Resource type: P

DB2 privileges

BIND

XAPLPRIV value: **BINDAUT**

Does the user own the plan?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLOWNQ parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.plan-name.BIND	MDSNPN or GDSNPN
DB2-subsystem.owner.BINDAGENT	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

EXECUTE

XAPLPRIV value: **CHKEXEC**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.plan-name.EXECUTE	MDSNPN or GDSNPN
DB2-subsystem.SYSADM	DSNADM

Schema privileges

Resources: Schemas

Resource type: M

DB2 privileges

ALTERIN

XAPLPRIV value: **ALTINAUT**

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the object?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
# *DB2-subsystem.schema-name*.ALTERIN	MDSNSC or GDSNSC
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CHANGE NAME QUALIFIER

XAPLPRIV value: **QUALAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Note: No RACF audit record or ICH408I message is generated for a failure related to this privilege. RACF will audit successes, if specified.

COMMENT ON

XAPLPRIV value: **COMNTAUT**

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the object?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
# *DB2-subsystem.schema-name*.ALTERIN	MDSNSC or GDSNSC
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CREATEIN

XAPLPRIV value: **CREINAUT**

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOBJN parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.schema-name.CREATEIN	MDSNSC or GDSNSC
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DROPIN

XAPLPRIV value: **DRPINAUT**

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the object?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
# *DB2-subsystem.schema-name.object-name*.DROPIN	MDSNSC
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Sequence privileges

Resources: Sequences

Resource type: Q

DB2 privileges

ALTER

XAPLPRIV value: **ALTERAUT**

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the sequence?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
# *DB2-subsystem.schema-name*.ALTERIN	MDSNSC or GDSNSC

One of these resources:	In class:
DB2-subsystem.schema-name.sequence-name.ALTER	MDSNSQ or GDSNSQ
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

COMMENT ON

XAPLPRIV value: **COMNTAUT**

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the sequence?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
# *DB2-subsystem.schema-name*.ALTERIN	MDSNSC or GDSNSC
DB2-subsystem.schema-name.sequence-name.ALTER	MDSNSQ or GDSNSQ
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

USAGE

XAPLPRIV value: **USAGEAUT**

Does the user own the sequence?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.schema-name.sequence-name.USAGE	MDSNSQ or GDSNSQ
DB2-subsystem.SYSADM	DSNADM

Storage group privileges

Resources: Storage groups

Resource type: S

DB2 privileges

DROP, ALTER

XAPLPRIV values: **DROPAUT, ALTERAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

USE

XAPLPRIV value: **USEAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.storage-groupname.USE	MDSNSG or GDSNSG
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Stored procedure privileges

Resources: Stored procedures

Resource type: O

DB2 privileges

DISPLAY

XAPLPRIV value: **DISPAUT**

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the stored procedure?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
# *DB2-subsystem.owner.object*.DISPLAY	MDSNSP
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

EXECUTE

XAPLPRIV value: **CHKEXEC**

Does the user own the stored procedure?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by
the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.schema-name.procedure-name.EXECUTE	MDSNSP or GDSNSP
DB2-subsystem.SYSADM	DSNADM

START

XAPLPRIV value: **STRTAUT**

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the stored procedure?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

STOP

XAPLPRIV value: **STPAUT**

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the stored procedure?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

System privileges

Resources: Systems

Resource type: U

DB2 administrative authorities

SYSADM

XAPLPRIV value: **SYSAAUTH**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSADM	DSNADM

SYSCTRL

XAPLPRIV value: **SYSCAUTH**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DB2 privileges

ALTER BUFFERPOOL

XAPLPRIV value: **CHKALTBP**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

BINDADD

XAPLPRIV value: **BINDAAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.BINDADD	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

BINDAGENT

XAPLPRIV value: **BNDAGAUT**

The user must have sufficient authority to:

One of these resources:	In class:
*DB2-subsystem.owner.*BINDAGENT	MDSNSM or GDSNSM
*DB2-subsystem.*SYSCTRL	DSNADM
*DB2-subsystem.*SYSADM	DSNADM

CANCEL | START | STOP DDF, DISPLAY | START | STOP RLIMIT

XAPLPRIV values: **CHKSTART, CHKSTOP, CHKDSPL, CHKDDF**

The user must have sufficient authority to:

One of these resources:	In class:
*DB2-subsystem.*SYSOPR	DSNADM
*DB2-subsystem.*SYSCTRL	DSNADM
*DB2-subsystem.*SYSADM	DSNADM

CREATEALIAS

XAPLPRIV value: **CRTALAUT**

The user must have sufficient authority to:

One of these resources:	In class:
*DB2-subsystem.*CREATEALIAS	MDSNSM or GDSNSM
*DB2-subsystem.*SYSCTRL	DSNADM
*DB2-subsystem.*SYSADM	DSNADM
Note: DBADM and DBCTRL authorities can be used to allow a user to create aliases. See “CREATE ALIAS privilege” on page 45 for more information.	
*DB2-subsystem.database-name.*DBCTRL	DSNADM
*DB2-subsystem.database-name.*DBADM	DSNADM

CREATEDBA

XAPLPRIV value: **CRTDBAUT**

The user must have sufficient authority to:

One of these resources:	In class:
*DB2-subsystem.*CREATEDBA	MDSNSM or GDSNSM
*DB2-subsystem.*CREATEDBC	MDSNSM or GDSNSM
*DB2-subsystem.*SYSCTRL	DSNADM
*DB2-subsystem.*SYSADM	DSNADM

CREATESG

XAPLPRIV value: **CRTSGAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.CREATESG	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CREATETMTAB

XAPLPRIV value: **CRTTMAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.CREATETMTAB	MDSNSM or GDSNSM
DB2-subsystem.CREATETAB	MDSNDB or GDSNDB
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DISPLAY, DISPLAY BUFFERPOOL

XAPLPRIV values: **CHKDISPL, CHKDSPBP**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.DISPLAY	MDSNSM or GDSNSM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DISPLAY ARCHIVE

XAPLPRIV value: **DARCHAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.DISPLAY	MDSNSM or GDSNSM
DB2-subsystem.ARCHIVE	MDSNSM or GDSNSM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

MONITOR1

XAPLPRIV value: **MON1AUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.MONITOR1	MDSNSM or GDSNSM

One of these resources:	In class:
DB2-subsystem.MONITOR2	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

MONITOR2

XAPLPRIV value: **MON2AUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.MONITOR2	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

RECOVER BSDS

XAPLPRIV value: **CHKBSDS**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.BSDS	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

RECOVER INDOUBT

XAPLPRIV value: **CHKRECOV**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.RECOVER	MDSNSM or GDSNSM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

SET ARCHIVE

XAPLPRIV value: **SARCHAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.ARCHIVE	MDSNSM or GDSNSM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

STOPALL
XAPLPRIV value: **CHKSUBSY**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.STOPALL	MDSNSM or GDSNSM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

STOSPACE UTILITY
XAPLPRIV value: **STOAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.STOSPACE	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

TRACE
XAPLPRIV value: **CHKTRACE**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.TRACE	MDSNSM or GDSNSM
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

USE ARCHIVE LOG
XAPLPRIV value: **ARCHAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.ARCHIVE	MDSNSM or GDSNSM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

Table privileges

Resources: Tables

Resource type: T

Note about SYSCTRL

The SYSCTRL administrative authority does not apply to user tables. DB2 turns on bit 7 of the XAPLFLG1 field for a user table. If this bit is on, the RACF access control module bypasses checking for the SYSCTRL authority. This allows RACF processing to model DB2 processing.

DB2 privileges

ALTER

XAPLPRIV value: **ALTERAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.table-owner.table-name.ALTER	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

ALTER INDEX, DROP INDEX

XAPLPRIV values: **ALTIXAUT, DRPIXAUT**

Does the user own the index?

If so, XAPLUPRM or XAPLUCHK must match the index name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CHANGE NAME QUALIFIER

XAPLPRIV value: **QUALAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM

One of these resources:	In class:
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

COMMENT ON, COMMENT ON INDEX, DROP

XAPLPRIV values: **COMNTAUT, CMTIXAUT, DROPAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

CREATE SYNONYM

XAPLPRIV value: **CRTSYAUT**

There are no authorization checks (return code 4).

CREATE VIEW

XAPLPRIV value: **CRTVUAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM
DB2-subsystem.DB2-database-name-1.DBADM *DB2-subsystem.DB2-database-name-2*.DBADM ⋮ *DB2-subsystem.DB2-database-name-n*.DBADM	DSNADM DSNADM ⋮ DSNADM

Note: DBADM authority can be used to allow a user to create views. See “CREATE VIEW privilege” on page 44 for more information.

DELETE

XAPLPRIV value: **DELETAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.table-owner.table-name.DELETE	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

DROP ALIAS

XAPLPRIV value: **DRPALAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DROP SYNONYM

XAPLPRIV value: **DRPSYAUT**

There are no authorization checks (return code 4).

INDEX

XAPLPRIV value: **INDEXAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.table-owner.table-name.INDEX	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

INSERT

XAPLPRIV value: **INSRTAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.table-owner.table-name.INSERT	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

LOAD

XAPLPRIV value: **LOADAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.LOAD	MDSNDB or GDSNDB
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

LOCK TABLE

XAPLPRIV value: **LOCKAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.table-owner.table-name.SELECT	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

REFERENCES

XAPLPRIV value: **REFERAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.table-owner.table-name.REFERENCES	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.ALTER	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.column.REFERENCES	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

REFRESH

XAPLPRIV value: **RFRSHAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

RENAME TABLE

XAPLPRIV value: **RNTABAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.DBMAINT	DSNADM
DB2-subsystem.database-name.DBCTRL	DSNADM
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

SELECT

XAPLPRIV value: **SELCTAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.table-owner.table-name.SELECT	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

TRIGGER

XAPLPRIV value: **TRIGAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.table-owner.table-name.TRIGGER	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.ALTER	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

UPDATE

XAPLPRIV value: **UPDTEAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.table-owner.table-name.UPDATE	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.column.UPDATE	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

Any of the table privileges

XAPLPRIV value: **ANYTBAUT**

Does the user own the table?

If so, XAPLUPRM or XAPLUCHK must match the table name qualifier passed from DB2.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.table-owner.table-name.REFERENCES	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.ALTER	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.INDEX	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.SELECT	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.INSERT	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.DELETE	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.UPDATE	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL This check is bypassed for user tables.	DSNADM
DB2-subsystem.SYSADM	DSNADM

Tablespace privileges

Resources: Tablespaces

Resource type: R

DB2 privileges

DROP, ALTER

XAPLPRIV values: **DROPAUT, ALTERAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

USE

XAPLPRIV value: **USEAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.database-name.tablespace-name.USE	MDSNTS or GDSNTS
DB2-subsystem.database-name.DBADM	DSNADM

One of these resources:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

User-defined distinct type privileges

Resources: User-defined distinct types

Resource type: E

DB2 privileges

USAGE

XAPLPRIV value: **USAGEAUT**

Does the user own the user-defined distinct type?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.schema-name.type-name.USAGE	MDSNUT or GDSNUT
DB2-subsystem.SYSADM	DSNADM

User-defined function privileges

Resources: User-defined functions

Resource type: F

DB2 privileges

DISPLAY

XAPLPRIV value: **DISPAUT**

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the user-defined function?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
# *DB2-subsystem.owner.object*.DISPLAY	MDSNUF
DB2-subsystem.SYSOPR	DSNADM

One of these resources:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

EXECUTE

XAPLPRIV value: **CHKEXEC**

Does the user own the user-defined function?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by
the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.schema-name.function-name.EXECUTE	MDSNUF or GDSNUF
DB2-subsystem.SYSADM	DSNADM

START

XAPLPRIV value: **STRTAUT**

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the user-defined function?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

STOP

XAPLPRIV value: **STPAUT**

Does the user match the schema name?

If so, XAPLUPRM or XAPLUCHK must match the schema name passed from DB2 by the XAPLOWNQ parameter.

If not, does the user own the user-defined function?

If so, XAPLUPRM or XAPLUCHK must match the owner name passed from DB2 by the XAPLREL1 parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSOPR	DSNADM
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

View privileges

Resources: Views

Resource type: V

DB2 privileges

COMMENT ON

XAPLPRIV value: **COMNTAUT**

Does the user own the view?

If so, XAPLUPRM or XAPLUCHK must match the view name passed from DB2 by the XAPLOWNQ parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

DELETE

XAPLPRIV value: **DELETAUT**

Is the view updatable (for example, a view created from a single table)?

If so, does the user own the table?

If not, the user must have sufficient authority to:

One of these resources:	In class:
# *DB2-subsystem.table-owner.table-name*.DELETE	MDSNTB or GDSNTB
# *DB2-subsystem.database-name*.DBADM	DSNADM
DB2-subsystem.SYSADM	DSNADM

Note: *table-owner*, *table-name*, and *database-name* are for the base table for the
view.

Is the view a read-only view (for example, created from multiple tables)?

If so, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.view-owner.view-name.DELETE	MDSNTB or GDSNTB
DB2-subsystem.SYSADM	DSNADM

DROP

XAPLPRIV value: **DROPAUT**

Does the user own the view?

If so, XAPLUPRM or XAPLUCHK must match the view name passed from DB2 by the XAPLOWNQ parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

INSERT

XAPLPRIV value: **INSRTAUT**

Is the view updatable (for example, a view created from a single table)?

If so, does the user own the table?

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.table-owner.table-name.INSERT	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSADM	DSNADM

Note: *table-owner*, *table-name*, and *database-name* are for the base table for the view.

Is the view a read-only view (for example, created from multiple tables)?

If so, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.view-owner.view-name.INSERT	MDSNTB or GDSNTB
DB2-subsystem.SYSADM	DSNADM

REGENERATE VIEW

XAPLPRIV value: **ALTERAUT**

Does the user own the view?

If so, XAPLUPRM or XAPLUCHK must match the view name passed from DB2 by the XAPLOWNQ parameter.

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.SYSCTRL	DSNADM
DB2-subsystem.SYSADM	DSNADM

SELECT

XAPLPRIV value: **SELCTAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.view-owner.view-name.SELECT	MDSNTB or GDSNTB
DB2-subsystem.SYSADM	DSNADM

UPDATE

XAPLPRIV value: **UPDTEAUT**

Is the view updatable (for example, a view created from a single table)?

If so, does the user own the table?

If not, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.table-owner.table-name.UPDATE	MDSNTB or GDSNTB
DB2-subsystem.table-owner.table-name.column-name.UPDATE	MDSNTB or GDSNTB
DB2-subsystem.database-name.DBADM	DSNADM
DB2-subsystem.SYSADM	DSNADM

Note: *table-owner*, *table-name*, *column-name*. and *database-name* are for the base
table for the view.

Is the view a read-only view (for example, created from multiple tables)?

If so, the user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.view-owner.view-name.UPDATE	MDSNTB or GDSNTB
DB2-subsystem.SYSADM	DSNADM

″Any table″ authority

XAPLPRIV value: **ANYTBAUT**

The user must have sufficient authority to:

One of these resources:	In class:
DB2-subsystem.view-owner.view-name.SELECT	MDSNTB or GDSNTB
DB2-subsystem.view-owner.view-name.INSERT	MDSNTB or GDSNTB
DB2-subsystem.view-owner.view-name.UPDATE	MDSNTB or GDSNTB
DB2-subsystem.view-owner.view-name.DELETE	MDSNTB or GDSNTB
DB2-subsystem.SYSCTRL This check is bypassed when bit 7 of XAPLFLG1 is on.	DSNADM
DB2-subsystem.SYSADM	DSNADM

Appendix E. DB2 RACF access control module messages

IRR900A RACF/DB2 EXTERNAL SECURITY MODULE FAILED TO INITIALIZE FOR DB2 SUBSYSTEM *subsystem-name* **BECAUSE CLASS** *classname* **COULD NOT BE RACLISTED. RACROUTE RETURN CODE** *return_code*, **RACF RETURN CODE** *return_code*, **REASON CODE** *reason_code*.

Explanation: The RACF access control module initialization function for DB2 subsystem *subsystem-name* attempted to RACLIST class *classname* using RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES. If this is DB2 data sharing, *subsystem-name* is the group attach name. Otherwise, it is the DB2 subsystem. The RACROUTE request failed with the return and reason codes provided in the message text. The return and reason codes are shown in hexadecimal format.

System action: See System Action for message IRR912I or IRR913I.

Operator response: Contact the system programmer.

System programmer response: Use the RACROUTE return code and RACF return and reason codes to determine the cause of the failure. After you correct the problem, restart DB2.

Destination: Descriptor code is 2. Routing codes are 1 and 9.

IRR901A RACF/DB2 EXTERNAL SECURITY MODULE FAILED TO INITIALIZE FOR DB2 SUBSYSTEM *subsystem-name* **BECAUSE NO ACTIVE DB2 RELATED CLASSES WERE FOUND.**

Explanation: The RACF access control module initialization function for subsystem *subsystem-name* determined that no classes for the indicated DB2 subsystem are active. If this is DB2 data sharing, *subsystem-name* is the group attach name. Otherwise, it is the DB2 subsystem.

System action: See System Action for message IRR912I or IRR913I.

Operator response: Contact your security administrator.

Security Administrator Response: Activate the desired classes for the indicated DB2 subsystem and restart DB2.

Destination: Descriptor code is 2. Routing codes are 1 and 9.

IRR902A RACF/DB2 EXTERNAL SECURITY MODULE FAILED TO INITIALIZE FOR DB2 SUBSYSTEM *subsystem-name* **BECAUSE THE INPUT ACEE WAS {MISSING | NOT VALID}.**

Explanation: The RACF access control module initialization function for subsystem *subsystem-name* determined that the input DB2 subsystem ACEE was either not valid or missing. If this is DB2 data sharing, *subsystem-name* is the group attach name. Otherwise, it is the DB2 subsystem.

System action: See System Action for message IRR912I or IRR913I.

Operator response: Contact the DB2 system programmer.

System programmer response: Contact the IBM support center.

Destination: Descriptor code is 2. Routing codes are 1 and 9.

IRR903A RACF/DB2 EXTERNAL SECURITY MODULE FAILED TO INITIALIZE FOR DB2 SUBSYSTEM *subsystem-name* **BECAUSE RACF WAS NOT ACTIVE.**

Explanation: The RACF access control module initialization function for subsystem *subsystem-name* determined that RACF is not active on this system. If this is DB2 data sharing, *subsystem-name* is the group attach name. Otherwise, it is the DB2 subsystem.

System action: See System Action for message IRR912I or IRR913I.

Operator response: Contact the RACF system programmer.

Problem determination: Issue the RVARY LIST command to determine RACF status.

System programmer response: Determine why RACF is inactive. After you correct the problem, activate RACF and restart DB2.

Destination: Descriptor code is 2. Routing codes are 1 and 9.

IRR904I RACF/DB2 EXTERNAL SECURITY MODULE INITIALIZED WITH WARNINGS FOR DB2 SUBSYSTEM *subsystem-name* **BECAUSE A DEFAULT ACEE COULD NOT BE CREATED. RACROUTE RETURN CODE** *return_code*, **RACF RETURN CODE** *return_code*, **REASON CODE** *reason_code*.

Explanation: The RACF access control module initialization function for subsystem *subsystem-name* attempted to create a default ACEE to use in subsequent authority checking when no ACEE is provided. If this is DB2 data sharing, *subsystem-name* is the group attach name. Otherwise, it is the DB2 subsystem.

The attempt to create the ACEE using RACROUTE REQUEST=VERIFY,ENVIR=CREATE failed with the return and reason codes provided in the message text. The return and reason codes are shown in hexadecimal format.

System action: Processing continues and the RACF access control module is used for subsequent authority checking if DB2 provides an ACEE. If no ACEE is provided, requests are deferred to DB2.

Operator response: Contact the DB2 system programmer.

System programmer response: Use the RACROUTE return code and RACF return and reason codes to determine the cause of the failure. After you correct the problem, restart DB2.

Destination: Descriptor code is 12. Routing codes are 2, 9, and 10.

IRR905I RACF/DB2 TERMINATION FUNCTION COMPLETED WITH WARNINGS FOR DB2 SUBSYSTEM *subsystem-name* **BECAUSE CLASS** *classname* **COULD NOT BE UN-RACLISTED. RACROUTE RETURN CODE** *return_code***, RACF RETURN CODE** *return_code***, REASON CODE** *reason_code***.**

Explanation: The RACF access control module termination function for subsystem *subsystem-name* attempted to delete RACLISTed profiles for class *classname*. If this is DB2 data sharing, *subsystem-name* is the group attach name. Otherwise, it is the DB2 subsystem.

The attempt to delete the profiles using RACROUTE REQUEST=LIST,ENVIR=DELETE failed with the return and reason codes provided in the message text. The return and reason codes are in hexadecimal format.

System action: The termination function continues processing. Resources are cleaned up when processing completes. This does not impact RACF authorization checking when DB2 is restarted.

Operator response: Contact the DB2 system programmer.

System programmer response: Use the RACROUTE return code and the RACF return and reason codes to determine the cause of the failure.

Destination: Descriptor code is 12. Routing codes are 2, 9, and 10.

IRR906I RACF/DB2 TERMINATION FUNCTION COMPLETED WITH WARNINGS FOR DB2 SUBSYSTEM *subsystem-name* **BECAUSE THE DEFAULT ACEE COULD NOT BE DELETED. RACROUTE RETURN CODE** *return_code***, RACF RETURN CODE** *return_code***, REASON CODE** *reason_code***.**

Explanation: The RACF access control module termination function for the subsystem *subsystem-name* attempted to delete the default ACEE used by the RACF access control module. If this is DB2 data sharing, *subsystem-name* is the group attach name. Otherwise, it is the DB2 subsystem.

The attempt to delete the ACEE using RACROUTE REQUEST=VERIFY,ENVIR=DELETE failed with the return and reason codes provided in the message text. The return and reason codes are in hexadecimal format.

System action: The termination function continues processing and resources are cleaned up when processing completes. This does not impact RACF authorization checking when DB2 is restarted.

Operator response: Contact the DB2 system programmer.

System programmer response: Use the RACROUTE return code and the RACF return and reason codes to determine the cause of the failure. After you correct the problem, restart DB2.

Destination: Descriptor code is 12. Routing codes are 2, 9, and 10.

IRR907I RACF/DB2 TERMINATION FUNCTION COMPLETED WITH WARNINGS FOR DB2 SUBSYSTEM *subsystem-name* **BECAUSE THE INPUT ACEE WAS {MISSING | NOT VALID}.**

Explanation: The RACF access control module termination function for the subsystem *subsystem-name* determined that the input DB2 subsystem ACEE was either not valid or missing. If this is DB2 data sharing, *subsystem-name* is the group attach name. Otherwise, it is the DB2 subsystem.

System action: For exit termination, the RACF access control module is not able to complete its termination function. This should not impact RACF authorization checking when DB2 is restarted.

Operator response: Contact the DB2 system programmer.

System programmer response: Contact the IBM support center.

Destination: Descriptor code is 12. Routing codes are 2, 9, and 10.

IRR908I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM *subsystem-name* **HAS A MODULE VERSION OF** *module-version* **AND A MODULE LENGTH OF** *module-length***.**

Explanation: The RACF access control module initialization function for subsystem *subsystem-name* has determined the version and length of the RACF access control module for subsystem *subsystem-name*. If this is DB2 data sharing, *subsystem-name* is the group attach name. Otherwise, it is the DB2 subsystem. *module-version* is the FMID or APAR number associated with the module. *module-length* is the hexadecimal length of all CSECTs contained in the module.

System action: The RACF access control module continues.

Destination: Descriptor code is 4. Routing codes are 9 and 10.

IRR909I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM *subsystem-name* **IS USING OPTIONS: &CLASSOPT=** *classopt* **&CLASSNMT=** *classnmt* **&CHAROPT=** *charopt* **&ERROROPT=** *erroropt* **&PCELLCT=** *pcellct* **&SCELLCT=** *scellct*

Explanation: The RACF access control module initialization function for subsystem *subsystem-name* lists the options that are being used for the RACF access control module. If this is DB2 data sharing, *subsystem-name* is the group attach name. Otherwise, it is the DB2 subsystem. For an explanation of the options, see *z/OS Security Server RACF System Programmer's Guide*.

System action: The RACF access control module continues.

Destination: Descriptor code is 4. Routing codes are 9 and 10.

IRR910I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM *subsystem-name* **INITIATED RACLIST FOR CLASSES: {***classname-list* **| * NONE *}**

Explanation: The RACF access control module initialization function for DB2 subsystem *subsystem-name* issued a RACROUTE REQUEST=LIST,GLOBAL=YES macro for classes *classname-list* as defined in the object table in the RACF access control module. If `* NONE *` is displayed, an error occurred before the initialization function could issue RACROUTE REQUEST=LIST for any class. If this is DB2 data sharing, *subsystem-name* is the group attach name. Otherwise, it is the DB2 subsystem.

System action: The RACF access control module continues.

Destination: Descriptor code is 4. Routing codes are 9 and 10.

IRR911I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM *subsystem-name* **SUCCESSFULLY RACLISTED CLASSES: {***classname-list* **| * NONE *}**

Explanation: The RACF access control module initialization function for DB2 subsystem *subsystem-name* lists the classes for which the RACROUTE REQUEST=LIST,GLOBAL=YES macro was successful. If `* NONE *` is displayed, no classes were RACLISTed successfully. See message IRR910I to determine which classes the RACF access control module attempted to use. The class list displayed in IRR911I might be a valid subset of the classes listed in message IRR910I. See *z/OS Security Server RACF Security Administrator's Guide* for more information about initializing the RACF access control module.

System action: The RACF access control module continues.

Destination: Descriptor code is 4. Routing codes are 9 and 10.

IRR912I NATIVE DB2 AUTHORIZATION IS USED.

Explanation: RACF is not being used to control access to DB2 resources. This message is preceded by other messages that describe why RACF is not being used for access control decisions.

System action: None. All subsequent access control decisions are made by DB2 using DB2's native security mechanism.

Operator response: Follow the Operator Response for the message that preceded this message.

Destination: Descriptor code is 2. Routing codes are 1 and 9.

IRR913I DB2 SUBSYSTEM TERMINATION REQUESTED.

Explanation: RACF has requested that the DB2 subsystem be terminated. This message is preceded by another message which describes why this request has been made.

System action: RACF has requested that the DB2 subsystem terminate.

Operator response: Follow the Operator Response for the message that preceded this message.

Destination: Descriptor code is 2. Routing codes are 1 and 9.

IRR914I DSNX@XAC has been invoked with a DB2 VxRxMx parameter list

Explanation: RACF access control module was invoked from a DB2 Version 8 system. However, the parameter list that was passed was for another version of DB2. This mismatch of DB2 version and level of the RACF access control module is not allowed.

System action: If RACF access control module has installation option `&ERROROPT 2` specified, then the DB2 subsystem is asked to terminate. If installation option `&ERROROPT 1` was specified, then the DB2 subsystem is asked to use native DB2 authorization. In either case, the exit is not called again.

System programmer response: DB2 Version 8 must be executed with the DSNX@XAC that was shipped with DB2 Version 8. The DB2 V8-shipped version must be assembled with the DB2 Version 8 macros, link edited, and installed in a library which is accessible to your DB2 subsystem. DB2 Version 7 and DB2 Version 6 must be executed with the RACF/DB2 external security module that was shipped by RACF in `SYS1.SAMPLIB(RACF/DB2 external security module)`. This code must be assembled with the DB2 macros of the correct DB2 release, link edited, and installed in a library which is accessible to your DB2 subsystem.

Destination: Descriptor code is 12. Routing codes are 2, 9, and 10.

IRR915I EXPLRC1 = ***xxx*****, EXPLRC2 =** ***xxx*****, XAPLPRIV =** ***xxxx***

Explanation: RACF access control module has been instructed (either by a zap or by changing the assembler source) to display the return and reason code (EXPLRC1 and EXPLRC2) that is returned to DB2 along with the DB2 privilege code (XAPLPRIV) for the request. For DB2 initialization and termination, XAPLPRIV will be *xxx*.

System action: None. This message is a diagnostic informational message.

System programmer response: None. This message is only issued if RACF access control module has been specifically altered to display the return, reason, and privilege codes. This should only be done under the guidance of the IBM service team.

Destination: Descriptor code is 4. Routing codes are 9 and 10.

Notices

Trademarks

Index

How to send your comments

Your feedback helps IBM to provide quality information. Please send any comments that you have about this book or other DB2 UDB for z/OS documentation. You can use the following methods to provide comments:

- Send your comments by e-mail to db2zinfo@us.ibm.com and include the name of the product, the version number of the product, and the number of the book. If you are commenting on specific text, please list the location of the text (for example, a chapter and section title or a help topic title).
- You can send comments from the Web. Visit the library Web site at:

  www.ibm.com/software/db2zos/library.html

  This Web site has a an online reader comment form that you can use to send comments.
- You can also send comments by using the feedback link at the footer of each page in the Information Management Software for z/OS Solutions Information Center at http://publib.boulder.ibm.com/infocenter/db2zhelp.

Program Number: 5625-DB2

Printed in USA

SC18-7433-05