Security Server
RACROUTE Macro Reference

Version 2 Release 1
Contents

Figures .................................. vii

Tables ................................... ix

About this document ...................... xi
Who should use this document ........... xi
How to use this document ............... xiv
Where to find more information ....... xiv
Other sources of information ......... xiv
How to send your comments to IBM ... xv
If you have a technical problem ....... xv

z/OS Version 2 Release 1 summary of changes ......................... xvii

Chapter 1. How to use the RACF system macros ....................... 1
Reading the macro instructions ........ 4
Continuation lines ....................... 6

Chapter 2. RACF system macros ............... 7
RACROUTE: Router interface ............ 8
Keyword and parameter cross-reference for RACROUTE .......... 9
Addressing considerations ............. 12
Cross memory considerations ........... 12
RACROUTE (standard form) ............. 13
Return codes .......................... 17
RACROUTE (list form) .................. 18
RACROUTE (execute form) .............. 19
RACROUTE (modify form) .............. 20

Chapter 3. System macros ................. 23
RACROUTE REQUEST=AUDIT: General-purpose security-audit request ....... 23
Return codes and reason codes ......... 26
RACROUTE REQUEST=AUDIT (list form) ....... 27
RACROUTE REQUEST=AUDIT (execute form) .. 28
RACROUTE REQUEST=AUDIT (modify form) ... 29
RACROUTE REQUEST=AUTH: Check RACF authorization ............. 30
Return codes and reason codes ......... 44
RACROUTE REQUEST=AUTH (list form) ....... 49
RACROUTE REQUEST=AUTH (execute form) .. 51
RACROUTE REQUEST=AUTH (modify form) ... 54
RACROUTE REQUEST=DEFINE: Define, modify, rename, or delete a resource for RACF .. 57
Return codes and reason codes ......... 73
RACROUTE REQUEST=DEFINE (list form) ....... 79
RACROUTE REQUEST=DEFINE (execute form) ... 81
RACROUTE REQUEST=DEFINE (modify form) ... 85
RACROUTE REQUEST=DIRAUTH: Directed authorization check of security classification ....... 87
Return codes and reason codes ......... 92
RACROUTE REQUEST=DIRAUTH (list form) ....... 95
RACROUTE REQUEST=DIRAUTH (execute form) .. 96
RACROUTE REQUEST=DIRAUTH (modify form) ... 97
RACROUTE REQUEST=EXTRACT: Replace or retrieve fields ................. 98
Return codes and reason codes ......... 119
RACROUTE REQUEST=EXTRACT (list form) ....... 129
RACROUTE REQUEST=EXTRACT (execute form) .. 131
RACROUTE REQUEST=EXTRACT (modify form) ... 133
RACROUTE REQUEST=FASTAUTH: Verify access to resources ................. 135
Return codes and reason codes ......... 144
RACROUTE REQUEST=FASTAUTH (list form) ....... 146
RACROUTE REQUEST=FASTAUTH (execute form) .. 147
RACROUTE REQUEST=FASTAUTH (modify form) ... 149
RACROUTE REQUEST=LIST: Build in-storage profiles ................. 148
RACROUTE REQUEST=LIST (standard form) ....... 149
Return codes and reason codes ......... 155
RACROUTE REQUEST=LIST (list form) ....... 158
RACROUTE REQUEST=LIST (execute form) .. 159
RACROUTE REQUEST=LIST (modify form) ... 160
RACROUTE REQUEST=SIGNON: Manage PV signed-on lists ................. 161
RACROUTE REQUEST=SIGNON (standard form) ....... 162
Return codes and reason codes ......... 169
RACROUTE REQUEST=SIGNON (list form) ....... 173
RACROUTE REQUEST=SIGNON (execute form) .. 175
RACROUTE REQUEST=SIGNON (modify form) ... 176
RACROUTE REQUEST=STAT: Determine RACF Status ................. 178
RACROUTE REQUEST=STAT (standard form) ....... 178
RACROUTE REQUEST=STAT (list form) ....... 181
RACROUTE REQUEST=STAT (execute form) .. 184
RACROUTE REQUEST=STAT (modify form) ... 185
RACROUTE REQUEST=TOKENBLD: Build a UTOKEN ................. 187
RACROUTE REQUEST=TOKENBLD (standard form) ....... 187
RACROUTE REQUEST=TOKENBLD (list form) ....... 191
RACROUTE REQUEST=TOKENBLD (execute form) .. 193
RACROUTE REQUEST=TOKENBLD (modify form) ... 194
RACROUTE REQUEST=TOKENMAP: Access token fields ................. 197
Appendix E. Supplied class descriptor table entries ........ 397

Appendix F. Requesting security services ........ 457

Appendix G. Accessibility ........ 459
Accessibility features ........ 459
Using assistive technologies ........ 459
Keyboard navigation of the user interface ........ 459

Dotted decimal syntax diagrams ........ 459

Notices ........ 463
Policy for unsupported hardware ........ 464
Minimum supported hardware ........ 465
Programming interface information ........ 465
RSA Secure code ........ 465
Trademarks ........ 465

Index ........ 467
Figures

1. Sample Macro Instruction ........ 4
2. Continuation Coding .......... 6
3. ENVR Data Structure .......... 109
4. ENVR data structure .......... 165
5. ENVR data structure .......... 215
## Tables

1. Cross-reference for RACROUTE REQUEST=type and the independent RACF system macros ........................................ 8
2. RACROUTE REQUEST= Operand cross-reference ............................................................................................................. 9
3. Types of profile checking performed by RACROUTE REQUEST=AUTH ................................................................. 40
4. Type of security label dominance required for successful authorization checking with each MAC type and ACCESS value ................................................................. 91
5. Description of ENVIR data structure ............................................................................................................................... 109
6. ENVROUT storage area processing ................................................................................................................................. 110
7. Description of ENVIR data structure ............................................................................................................................... 165
8. ENVROUT Storage Area Processing ............................................................................................................................... 165
9. Delete Actions Based on Supplied Input ............................................................................................................................. 168
10. Alphabetic sort order for NEXT= processing .................................................................................................................. 181
11. Description of ENVIR data structure ............................................................................................................................... 215
12. ENVROUT storage area processing ................................................................................................................................. 215
13. Relationship between the ERROROPT keyword and RELEASE= values ........................................................................ 217
14. Description of X500NAME data structure ........................................................................................................................... 231
15. Relationship between the ERROROPT keyword and RELEASE= values ........................................................................ 249
16. Cross-reference for RACROUTE REQUEST=type and the independent RACF system macros ........................................ 269
17. RACDEF parameters for RELEASE= 1.6 through 1.8.1 ......................................................................................... 287
18. Types of profile checking performed by RACHECK ........................................................................................................ 301
19. RACHECK parameters for RELEASE=1.6 through 1.8.2 ............................................................................................ 303
20. RACINIT parameters for RELEASE=1.6 through 1.8.1 .............................................................................................. 315
21. RACLST parameters for RELEASE=1.6 through 1.8.1 ............................................................................................... 323
22. RACSTAT parameters for RELEASE=1.6 through 1.8.1 ............................................................................................. 328
23. RACXTRT parameters for RELEASE=1.6 through 1.8.1 ............................................................................................ 340
24. RACROUTE macro keywords and their request-specific parameter lists ............................................................... 393
25. Classes supplied by IBM .................................................................................................................................................. 397
About this document

This document contains information about RACF®, which is part of z/OS Security Server, and supports z/OS Version 2 (5650-ZOS). It describes the full-function RACROUTE macro for z/OS®, and the requests that can be invoked by it, including syntax and related information.

Who should use this document

This document is intended (1) for the use of programmers who are writing applications that need to invoke RACF (or another external security product) from MVS™ and (2) for those who write other external security products (that replace RACF) to perform the following functions for MVS:

- Centralized auditing
- Resource authorization
- Resource definition
- Data encryption
- User identification and verification.

How to use this document

The major sections of this document contain information about the RACROUTE macro. The appendices present interface information. Macro descriptions within the sections are presented in alphabetical order. Each macro description includes:

- A general description of the service that the macro performs
- A table of syntax rules that you must follow when you code the macro
- An alphabetical list of the parameters you can specify and an explanation of each parameter:

These are the major sections and appendices in this document:

- The section Chapter 1, “How to use the RACF system macros,” on page 1 provides information about how to read the syntax tables and how to code the macros.
- The section Chapter 2, “RACF system macros,” on page 7 provides information about the external system macros that callers can use to invoke RACF or another security product. This section includes information on the full-function RACROUTE interface and the macro types that it can invoke.

  Guideline: Use the full-function RACROUTE interface documented in this chapter instead of the independent RACF system macros described in Appendix A, “Independent RACF system macros,” on page 269.

- The section Appendix A, “Independent RACF system macros,” on page 269 describes macros that can be used to invoke security checking by either RACF or another security product. For RACF 1.9 and later releases, no new keywords are supported on the independent RACF system macro invocations. However, IBM® continues to support the independent invocation of these macros for release levels prior to 1.9. If your installation wants to invoke these macros and use new keywords, your installation must invoke them using the full-function RACROUTE interface documented in Chapter 2, “RACF system macros,” on page 7:
Preface

- The section Appendix B, “RACF database templates,” on page 347 describes the Group, User, Connect, Data Set, General, and Reserved templates for the RACF database.
- The section Appendix C, “System authorization facility (SAF) and SAF exits,” on page 383 provides information about the System Authorization Facility (SAF), and SAF exits.
- The section Appendix D, “SAF interface to an external security product,” on page 391 provides programming information about using the SAF interface with an external security product other than RACF.
- The section Appendix E, “Supplied class descriptor table entries,” on page 397 lists the class entries supplied by IBM in the class descriptor table (ICHRRCDX).
- The section Appendix F, “Requesting security services,” on page 457 provides guidance information for application programmers requesting the services of the security product.
- The section “Appendix G. Accessibility,” contains product accessibility information.
- The section “Notices” on page 463 contains copyright and trademark information.

Where to find more information

When possible, this information uses cross-document links that go directly to the topic in reference using shortened versions of the document title. For complete titles and order numbers of the documents for all products that are part of z/OS, see z/OS Information Roadmap.

To find the complete z/OS library, including the z/OS Information Center, see z/OS Internet Library (http://www.ibm.com/systems/z/os/zos/bkserv/).

RACF courses

The following RACF classroom courses are available in the United States:

- ES191 Basics of z/OS RACF Administration
- BE870 Effective RACF Administration
- ES885 Exploiting the Advanced Features of RACF

IBM provides various educational offerings for RACF. For more information about classroom courses and other offerings, do any of the following:
- See your IBM representative
- Call 1-800-IBM-TEACH (1-800-426-8322)

Other sources of information

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

Internet sources

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

- Online library
  To view and print online versions of the z/OS publications, use this address: http://www.ibm.com/systems/z/os/zos/bkserv/
- Redbooks®
Preface

The documents known as IBM Redbooks that are produced by the International Technical Support Organization (ITSO) are available at the following address:

http://www.redbooks.ibm.com

• Enterprise systems security

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

http://www.ibm.com/systems/z/advantages/security/

• RACF home page

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

http://www.ibm.com/systems/z/os/zos/features/racf/

• RACF-L discussion list

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

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

listserv@listserv.uga.edu

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

subscribe racf-l first_name last_name

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

racf-l@listserv.uga.edu

• Sample code

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

To access this code from a Web browser, go to the RACF home page and select the “Resources” file tab, then select “Downloads” from the list, or go to http://www-03.ibm.com/systems/z/os/zos/features/racf/goodies.html.

The code is also available from ftp.software.ibm.com through anonymous FTP.

To get access:

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

   cd eserver/zseries/zos/racf/

An announcement will be posted on the RACF-L discussion list whenever something is added.

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

− Try to get access by using a Web browser and the links from the RACF home page.

− Use a different FTP client. If necessary, use a client that is based on command line interfaces instead of graphical interfaces.

− If your FTP client has configuration parameters for the type of remote system, configure it as UNIX instead of MVS.
Restrictions

Because the sample code and tools are not officially supported,
• There are no guaranteed enhancements.
• No APARs can be accepted.
How to send your comments to IBM

We appreciate your input on this publication. Feel free to comment on the clarity, accuracy, and completeness of the information or provide any other feedback that you have.

Use one of the following methods to send your comments:
1. Send an email to mhvrdfs@us.ibm.com.
2. Send an email from the 'Contact us' web page for z/OS (http://www.ibm.com/systems/z/os/zos/webqs.html).
3. Mail the comments to the following address:
   IBM Corporation
   Attention: MHVRCFS Reader Comments
   Department H6MA, Building 707
   2455 South Road
   Poughkeepsie, NY 12601-5400
   US
4. Fax the comments to us, as follows:
   From the United States and Canada: 1+845+432-9405
   From all other countries: Your international access code +1+845+432-9405

Include the following information:
• Your name and address.
• Your email address.
• Your telephone or fax number.
• The publication title and order number:
  z/OS V2R1.0 Security Server RACROUTE Macro Reference
  SA23-2294-00
• The topic and page number that is related to your comment.
• The text of your comment.

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

IBM or any other organizations use the personal information that you supply to contact you only about the issues that you submit.

If you have a technical problem

Do not use the feedback methods that are listed for sending comments. Instead, take one of the following actions:
• Contact your IBM service representative.
• Call IBM technical support.
• Visit the IBM Support Portal at z/OS support page (http://www.ibm.com/systems/z/support/)
z/OS Version 2 Release 1 summary of changes

See the following publications for all enhancements to z/OS Version 2 Release 1 (V2R1):

- z/OS Migration
- z/OS Planning for Installation
- z/OS Summary of Message and Interface Changes
- z/OS Introduction and Release Guide
Chapter 1. How to use the RACF system macros

There are four different forms of the RACF macros: standard (S), list (L), execute (E), and modify (M). An explanation of when and why to use each form follows.

- **Standard form (MF=S):**
  Use the standard form of the macro when writing your own user programs (such as a non-system module), programs that you store in your own loadlib. By design, the standard form of the macro generates an inline parameter list and then modifies it. Do not use the standard form of the macro to write reentrant code, because reentrant code cannot be modified. With few exceptions, if you use the standard form of the macro in writing reentrant code, the execution of the code results in an abend.

  The standard form of the macro does three things: It obtains storage, fills in the parameters you have specified on the parameter list, and generates a call to the service routine.

- **List, execute, and modify forms:**
  Use the list, execute, and modify forms of the macro in combination when you write a reenrant program or plan to have numerous invocations of the macro. The forms of the macro work together in the following way:

  - **List form (MF=L):**
    The list form is used in two ways: 1) to allocate storage in your program's dynamic area (DSECT), and 2) to provide a template in the control section (CSECT) from which the dynamic storage parameter list can be initialized. This implies that two list forms are generally used in one program. One is used to allocate storage, and the other is used to initialize that storage. For example, the parameter list length is copied from the control section parameter list to the dynamic storage parameter list. To ensure that a valid dynamic storage parameter list has been built, the entire parameter list residing in the control section should then be copied to the parameter list residing in the dynamic storage.

    Since parameter list lengths can change from one release of RACF to the next, it is important to specify the same release on all invocations of the macro whether they be list, modify, or execute forms. If you were to code RELEASE=7705 on the control section parameter list and RELEASE=2.4 on the dynamic storage parameter list, the copy could result in an abend, or the call to the service would yield unpredictable results, since the complete parameter list was not copied.

    **Note:** The expansion of the list form does not contain any executable instructions; therefore, you cannot use registers in the list form.

  - **Execute form (MF=E):**
    When you specify the execute form of the macro, you can change the initial parameters you specified on the list form of the macro. You can also specify additional allowable parameters you might not have specified on the list form. When you issue the execute form of the macro, you generate a call to the service routine. You can change the parameters on the macro with each subsequent invocation of the execute form of the macro.

  - **Modify form (MF=M):**
    When you specify the modify form of the macro, you, in effect modify the parameter list of the list form of the macro. When you set the parameters that
you want using the modify form of the macro, you can then use the execute form. The advantage of using the modify form is that it allows you to set only those parameters that you need. Thus, you can code a series of modify forms, followed by one execute form instead of many execute forms. This results in reducing the number of macro invocations that you need to code.

Note: You must use the list form of the macro in conjunction with the execute or modify forms. The list form initializes certain fields that the execute and modify forms do not modify. Also, you must be sure to specify the same values for RELEASE= and REQUEST= on the execute, list, and modify forms.

Following is a representation of the relationship between the list and execute forms of the RACROUTE macro. For more information, refer to z/OS MVS Programming: Assembler Services Guide.

---------------------------------------------------------------------
* Example of list and execute in a reentrant module.
*  **************************************************
* RACROUTE START
*  
*  BALR 12,0 Refer to linkage conventions
*  USING *,12 in z/OS MVS Programming:
*  Assembler Services Guide
*  USING DYNDAT,13
*  
*  **************************************************
*  Copy the static RACROUTE parameter list to the dynamic storage
*  parameter list.
*  **************************************************
*  LA 8,RACROUD Load the address of the dynamic storage parameter list.
*  
*  LA 10,RACROUS Load the address of the static parameter list.
*  
*  L  9,RACROUL Load the length of the parameter list.
*  
*  LR 11,9 Copy the length into register 11 for the MVCL.
*  
*  MVCL 8,10 Copy the static parameter list into the dynamically allocated storage.
*  
*  **************************************************
*  Establish addressability to RACROUTE parameters.
*  **************************************************
*  LA 3,TOKNOUT
*  USING TOKEN,3
*  MVI TOKLEN,TOKCURLN Initialize the TOKNOUT area with the length.
*  
*  MVI TOKVERS,TOKVER01 Initialize the TOKNOUT area
RACF macros are assembler macros; therefore you must invoke them in assembler statements. When you code a macro instruction, the assembler processes it by using the macro definitions supplied by IBM and placed in the macro library when the system is generated.

The assembler expands the macro instruction into executable machine instructions or data fields that are in the form of assembler-language statements, or both. The machine instructions branch around the data fields, load registers, and give control to the system. The instruction that gives control to the system for RACROUTE is a branch instruction. The macro expansion appears as part of the assembler output listing.
RACF system macros

Note: High Level Assembler or Assembler H is required to assemble the RACROUTE macros.

The data fields, which are derived from parameters of the macro instruction, are used at execution time by the control program routine that performs the service associated with the macro.

Reading the macro instructions

Each macro description begins with a syntax diagram.

The syntax layout assumes that the standard begin, end, and continue columns are used. Therefore, column 1 is assumed to be the begin column. To change the begin, end, and continue columns, use the ICTL instruction to establish the coding format you want to use. If you do not use ICTL, the assembler recognizes the standard columns. To code the ICTL instruction, see [HLASM Language Reference].

Figure 1 shows a sample macro instruction, RACROUTE REQUEST=AUDIT, and summarizes all the coding information that is available for it. The example is divided into three columns, A, B, and C.

![Diagram](https://via.placeholder.com/150)

- The first column A contains those parameters that are required for that macro instruction. If a single line appears in that column A1, the parameter on that line is required and you must code it. If two or more lines appear together A2, you must code the parameter appearing on one and only one of the lines.
RACF system macros

- The second column contains those parameters that are optional for that macro instruction. If a single line appears in that column, the parameter on that line is optional. If two or more lines appear together, the entire parameter is optional, but if you elect to make an entry, code one and only one of the lines.

- The third column provides additional information about coding the macro instruction.

When substitution of a variable is required in column, the following classifications are used:

**symbol**

Any symbol valid in the assembler language. That is, an alphabetic character followed by 0-7 alphameric characters, with no special characters and no blanks.

**Rx-type address**

Any address that is valid in an Rx-type instruction (such as LA).

**register (2) - (12)**

One of general registers 2 through 12, specified within parentheses, previously loaded with the right-adjusted value or address indicated in the parameter description. You must set the unused high-order bits to zero. You can designate the register symbolically or with an absolute expression.

**decimal digit**

Any decimal digit up to the value indicated in the parameter description. If both symbol and decimal digit are indicated, an absolute expression is also allowed.

**register (1)**

General register 1, previously loaded with the right-adjusted value or address indicated in the parameter description. You must set the unused high-order bits to zero. Designate the register as (1) only.

**A-type address**

Any address that can be written in an A-type address constant.

**default**

A value that is used in default of a specified value; that is, the value the system assumes if the parameter is not coded.

Use the parameters to specify the services and options to be performed, and write them according to the following rules:

- If the selected parameter is written in all capital letters (for example, PROFILE or ENTITY or ENTITYX), code the parameter exactly as shown.
- If the selected parameter is written in italics, substitute the indicated value, address, or name.
- If the selected parameter is a combination of capital letters and italics separated by an equal sign (for example, VOLSER=vol addr), code the capital letters and equal sign as shown, and then make the indicated substitution for the italics.
- Code commas and parentheses exactly as shown.
- Positional parameters (parameters without equal signs) appear first; you must code them in the order shown. You can code keyword parameters (parameters with equal signs) in any order.
- If you select a parameter, read the third column before proceeding to the next parameter. The third column often contains coding restrictions for the parameter.
Continuation lines

You can continue the parameter field of a macro instruction on one or more additional lines according to the following rules:

1. Enter a continuation character (not blank, and not part of the parameter coding) in column 72 of the line.
2. Continue the parameter field on the next line, starting in column 16. All columns to the left of column 16 must be blank.

You can code the parameter field being continued in one of two ways. Either code the parameter field through column 71, with no blanks, and continue in column 16 of the next line, or truncate the parameter field with a comma, where a comma normally falls, with at least one blank before column 71, and then continue in column 16 of the next line. Figure 2 shows an example of each method.

```
NAME1  OP1  OPERAND1,OPERAND2,OPERAND3,OPERAND4,OPERAND5,OPERAND6,OPX
NAME2  OP2  OPERAND1,OPERAND2,OPERAND3,OPERAND4,OPERAND5,OPERAND6,OPERAND7
```

Figure 2. Continuation Coding
Chapter 2. RACF system macros

This section contains the external RACF system macros that other callers can use to invoke RACF or another security product.

The RACF system macros are received as part of the MVS program product; installations receive these macros even if they do not intend to install RACF. The RACROUTE macro instruction is the interface for all products that provide resource control.

The following lists the RACF macros that you can invoke with the full function RACROUTE interface. IBM recommends that installations use the full function RACROUTE interface instead of the independent RACF system macros. Many of the keywords and macro invocations are supported only if you invoke them using this RACROUTE interface.

- “RACROUTE REQUEST=AUDIT: General-purpose security-audit request” on page 23 is used to audit requests to use a function or access a resource without authorization checking.
- “RACROUTE REQUEST=AUTH: Check RACF authorization” on page 30 is used to provide authorization checking when a user requests to use a function or access a resource.
- “RACROUTE REQUEST=DEFINE: Define, modify, rename, or delete a resource for RACF” on page 57 is used to define, modify, or delete resource profiles for RACF.
- “RACROUTE REQUEST=DIRAUTH: Directed authorization check of security classification” on page 87 is used to perform security label authorization checking for installations using security labels.
- “RACROUTE REQUEST=EXTRACT: Replace or retrieve fields” on page 98 is used to retrieve or update specified resource profile fields, to encode data, or to create an ENVR object, representing the security environment, from an existing ACEE.
- “RACROUTE REQUEST=FASTAUTH: Verify access to resources” on page 135 is used to provide authorization checking when a user requests access to a RACF-protected resource similar to RACROUTE REQUEST=AUTH. However, RACROUTE REQUEST=FASTAUTH verifies access to resources that have RACF profiles brought into main storage.
- “RACROUTE REQUEST=LIST: Build in-storage profiles” on page 148 is used to retrieve general resource profiles and build an in-storage list for faster authorization checking. The list is attached to the ACEE.
- “RACROUTE REQUEST=SIGNON: Manage PV signed-on lists” on page 161 is used to allow RACF to manage the signed-on lists associated with persistent verification.
- “RACROUTE REQUEST=STAT: Determine RACF Status” on page 178 is used to determine if RACF or another security product is active and, optionally, to determine whether protection is in effect for a given resource class. REQUEST=STAT can also be used to determine if a resource class name is defined.
- “RACROUTE REQUEST=TOKENBLD: Build a UTOKEN” on page 187 is used to modify an existing token.
RACROUTE macro

- "RACROUTE REQUEST=TOKENMAP: Access token fields" on page 197 is used to convert a user token (UTOKEN) or a resource token (RTOKEN) into either internal or external format.
- "RACROUTE REQUEST=TOKENXTR: Extract UTOKENs" on page 201 is used to extract a UTOKEN from the current task or address space ACEE.
- "RACROUTE REQUEST=VERIFY: Identify and verify a RACF-defined user" on page 206 is used to provide user identification and verification.
- "RACROUTE REQUEST=VERIFYX: Verify user and return a UTOKEN" on page 245 is used to create a user token (UTOKEN) for a unit of work. It provides for propagation of USERID, GROUPID, and SECLABEL for locally submitted jobs and is similar to VERIFY in some respects.

RACROUTE: Router interface

The RACROUTE macro is the interface to RACF (or another external security manager) for MVS resource managers. The macro descriptions in this document define this interface. This does not imply that the MVS operating system supports all the functions allowed by the interface. Rather, it defines the macros and keywords that are available for MVS resource managers to implement security for data and other resources.

The RACROUTE macro invokes the system authorization facility (SAF) router. If RACF is present, the SAF router directs control to the RACF router. If RACF is active, the RACF router then invokes RACF.

Note: High Level Assembler or Assembler H is required to assemble the RACROUTE macros.

In coding the RACROUTE macro to perform a particular request, you must also use the necessary parameters for that request type on the RACROUTE macro instruction. For example, if you code RACROUTE to access REQUEST=AUTH, you must code REQUEST=AUTH and any other required parameters as well as any optional ones you need from the RACROUTE REQUEST=AUTH macro. RACROUTE ensures that only the parameters applicable to the RACROUTE REQUEST=AUTH request type have been coded.

Note: When the function verifies the parameter list, if a keyword other than SEGDATA, STOKEN, TOKNIN, or TOKNOUT has an associated length and this length is zero, the function assumes that the keyword is not specified. For the SEGDATA keyword on the RACROUTE REQUEST=EXTRACT, TYPE=REPLACE, a length of zero is valid. For STOKEN, TOKNIN, and TOKNOUT, the keyword is considered not specified if both the associated length and version fields are zero in the token area.

Table 1 is a quick reference that identifies the system macro-request types that are replacements for the independent RACF system macros documented in Appendix A, “Independent RACF system macros,” on page 269.

<table>
<thead>
<tr>
<th>RACROUTE request type</th>
<th>Equivalent independent RACF system macro</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQUEST=AUTH</td>
<td>RACHECK</td>
</tr>
<tr>
<td>REQUEST=DEFINE</td>
<td>RACDEF</td>
</tr>
<tr>
<td>REQUEST=EXTRACT</td>
<td>RACXTRT</td>
</tr>
</tbody>
</table>
### Table 1. Cross-reference for RACROUTE REQUEST=type and the independent RACF system macros (continued)

<table>
<thead>
<tr>
<th>RACROUTE request type</th>
<th>Equivalent independent RACF system macro</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQUEST=FASTAUTH</td>
<td>FRACHECK</td>
</tr>
<tr>
<td>REQUEST=LIST</td>
<td>RACLST</td>
</tr>
<tr>
<td>REQUEST=STAT</td>
<td>RACSTAT</td>
</tr>
<tr>
<td>REQUEST=VERIFY</td>
<td>RACINIT</td>
</tr>
</tbody>
</table>

### Keyword and parameter cross-reference for RACROUTE

Table 2 lists the parameters available through the RACROUTE macro interface and cross-references them to each REQUEST=type in the entire RACROUTE macro. The allowable parameters for each REQUEST=type are marked with the symbol X. See the specific RACROUTE REQUEST=type macro descriptions for information about the use of the parameters for each type of request.

**Table 2. RACROUTE REQUEST= Operand cross-reference**

<table>
<thead>
<tr>
<th>RACROUTE parameters</th>
<th>AUDIT</th>
<th>AUTHOR</th>
<th>DEFINE</th>
<th>DIRAUTH</th>
<th>EXTRACT</th>
<th>FASTAUTH</th>
<th>LIST</th>
<th>SIGNON</th>
<th>STAT</th>
<th>TOKEN</th>
<th>VERIFY</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCLVL</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACEE</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACCEALET</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACCESS</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACTINFO</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>APPL</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATTR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUDIT</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUTHCHKS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRANCH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHKAUTH</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLASS</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COPY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COPYLEN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRITERIA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATA</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATEFMT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECOUPL</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DERIVE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSTYPE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENCRYPT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENTITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENTITYX</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENTRY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENVIR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Chapter 2. RACF system macros 9
<table>
<thead>
<tr>
<th>RACROUTE parameters</th>
<th>AUDIT</th>
<th>AUTH</th>
<th>DEFINE</th>
<th>DTRAUTH</th>
<th>FASTAUTH</th>
<th>L LIST</th>
<th>SIGNON</th>
<th>STAT</th>
<th>TOKENBLD</th>
<th>TOKENXR</th>
<th>VERIFY</th>
<th>VRFY</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENVRIN=</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENVROUT=</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERASE=</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERROROPT=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>EVENT=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EVQUAL=</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXENODE=</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>EXPDT=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXPDXT=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FIELDS=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FILESEQ=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FILTER=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLDACC=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FORMOUT=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GENERIC=</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>GLOBAL=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GROUP=</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>GROUPID=</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICRX=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICTX=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDID=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INSTLN=</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JOBNAME=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LEVEL=</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LIST=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOC=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOG=</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOGSTR=</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSTTYPE=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>MATCHGN=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCLASS=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MENTITY=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MENTX=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MF=</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>MGENER=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MGMTCLA=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSGRTRN=</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSGSP=</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSGSUPP=</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MVOLSER=</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 2. RACROUTE REQUEST= Operand cross-reference (continued)

| RACROUTE parameters | AUDIT | AUTH | DEFINE | DISTRUTH | EXTRACT | FASTAUTH | LIST | SIGNON | STAT | TOKENMAP | TOKENXTR | VERIFY | VRFY |
|---------------------|-------|------|--------|----------|---------|---------|------|--------|------|----------|----------|--------|
| NESTED=             |       |      |        |          |         |         |      |        |      |          |          |        | X    |
| NEWNAME=            | X     |      |        |          |         |         |      |        |      |          |          |        |      |
| NEWNAMX=            | X     |      |        |          |         |         |      |        |      |          |          |        |      |
| NEWPASS=            |       | X    |        |          |         |         |      |        |      |          |          |        |      |
| NEWPHRASE=          |       |      |        |          |         |         |      |        |      |          |          |        |      |
| NEXT=               |       |      |        |          |         |         |      |        |      |          |          |        |      |
| NOTIFY=             | X     |      |        |          |         |         |      |        |      |          |          |        |      |
| OIDCARD=            |       |      |        |          |         |         |      |        |      |          |          |        | X    |
| OLDVOL=             | X     |      |        |          |         |         |      |        |      |          |          |        |      |
| OWNER=              | X     | X    |        |          |         |         |      |        |      |          |          |        |      |
| PASSCHK=            |       |      |        |          |         |         |      |        |      |          |          |        | X    |
| PASSWRD=            |       |      |        |          |         |         |      |        |      |          |          |        | X    |
| PGMNAME=            |       |      |        |          |         |         |      |        |      |          |          |        | X    |
| PHRASE=             |       |      |        |          |         |         |      |        |      |          |          |        | X    |
| POE=                |       |      |        |          |         |         |      |        |      |          |          |        | X    |
| POENET=             |       |      |        |          |         |         |      |        |      |          |          |        | X    |
| PROFILE=            | X     |      |        |          |         |         |      |        |      |          |          |        |      |
| RACFIND=            | X     | X    |        |          |         |         |      |        |      |          |          |        |      |
| RECVR=              |       |      |        |          |         |         |      |        |      |          |          |        |      |
| RELATED=            | X     | X    | X       | X         | X        | X       | X     | X       | X     | X        | X        |        |      |
| RELEASE=            | X     | X    | X       | X         | X        | X       | X     | X       | X     | X        | X        |        |      |
| REMOTE=             |       |      |        |          |         |         |      |        |      |          |          |        | X    |
| REQSTOR=            | X     | X    | X       | X         | X        | X       | X     | X       | X     | X        | X        |        |      |
| RESCSECLABEL=       |       |      |        |          |         |         |      |        |      | X        | X        |        |      |
| RESOWN=             |       |      |        |          |         |         |      |        |      | X        | X        |        |      |
| RESULT=             | X     |      |        |          |         |         |      |        |      | X        | X        |        |      |
| RETPD=              |       |      |        |          |         |         |      |        |      | X        | X        |        |      |
| RTOKEN=             |       |      |        |          |         |         |      |        |      | X        | X        |        |      |
| SECLABEL=           | X     |      | X       | X         | X        | X       | X     | X       | X     | X        | X        |        |      |
| SECLVL=             | X     |      |        |          |         |         |      |        |      | X        | X        |        |      |
| SEGDATA=            | X     |      |        |          |         |         |      |        |      | X        | X        |        |      |
| SEGMENT=            | X     |      |        |          |         |         |      |        |      | X        | X        |        |      |
| SERVAUTH=           |       |      |        |          |         |         |      |        |      | X        | X        |        |      |
| SESSION=            | X     | X    | X       | X         | X        | X       | X     | X       | X     | X        | X        |        |      |
| SGROUP=             | X     | X    | X       | X         | X        | X       | X     | X       | X     | X        | X        |        |      |
| SMC=                |       |      |        |          |         |         |      |        |      | X        | X        |        |      |
| SNODE=              |       |      |        |          |         |         |      |        |      | X        | X        |        |      |
| START=              | X     |      |        |          |         |         |      |        |      | X        | X        |        |      |
| STAT=               | X     |      |        |          |         |         |      |        |      | X        | X        |        |      |
| STATUS=             | X     |      |        |          |         |         |      |        |      | X        | X        |        |      |
Addressing considerations

If a caller is executing in 24-bit addressing mode, all parameters and parameter lists are assumed to reside below 16MB. If a caller, however, is executing in 31-bit addressing mode, all parameters and parameter lists might reside above 16MB (that is, all parameter addresses are 31-bit addresses).

All parameter lists generated by the RACROUTE macro are in a format that allows assembled code to be moved above 16MB without being reassembled.

Cross memory considerations

Except where specifically noted, RACROUTE can only be issued in non-cross-memory mode (for example, where HASN=PASN=SASN). For the services that can be invoked in cross-memory mode (where it is not the case that HASN=PASN=SASN), RACROUTE must be issued in primary ASC mode.
RACROUTE (standard form)

The standard form of the RACROUTE macro is written as follows:

```
name name

name: Symbol. Begin name in column 1.
```

```
 b

One or more blanks must precede RACROUTE.
```

```
RACROUTE

RACROUTE

One or more blanks must follow RACROUTE.
```

```
REQUEST=type

REQUEST=type: System macro request type
```

```
,WORKA=work area addr

,WORKA=work area addr: A-type address or register (2) – (12)
```

```
,DECOUPL=YES
,DECOUPL=NO

Default: DECOUPL=NO
```

```
,MSGRTRN=YES
,MSGRTRN=NO

Default: MSGRTRN=NO
```

```
,MSGSP=subpool number

,MSGSP=subpool number: Decimal digit 0–255; default is 0.
```

```
,MSGSUPP=YES
,MSGSUPP=NO

Default: MSGSUPP=NO
```

```
,RELATED=value

,RELATED=value: Any valid macro keyword specified
```

```
,RELEASE=number

,RELEASE=number: 7790, 7780, 7770, 7760, 7750, 7740, 7730, 7720, 7709, 7708, 7707, 7706, 7705, 7703, 2608, 2.6, 2.4, 2.3, 2.2, 2.1, 1.9, 1.8.1, 1.8, 1.7 or 1.6

Note:
1. Default: RELEASE=1.6
2. Because the format of the FMID has changed, the release ID number for release 2.8 is 2608 and the release 10 number is 7703.
```

```
,REQSTOR=reqstor addr

,REQSTOR=reqstor addr: A-type address or register (2) – (12)
```

```
,SUBSYS=subsys addr

,SUBSYS=subsys addr: A-type address or register (2) – (12)
```

```
MF=S
```

For RACROUTE to work correctly, once you have chosen a REQUEST, you must also specify the parameters that belong to that request. See the RACROUTE REQUEST= macros for the necessary parameters.

This request requires a standard 18-word save area that is pointed to by register 13.

Data areas returned to the caller by RACF are either above or below 16 MB, depending upon the caller's addressing mode and the data area in question.

The parameters are explained as follows:

```
,REQUEST=AUDIT
```
specifies the system macro request type.

To invoke a system macro request supported through the RACROUTE interface, you must also code the parameters associated with that particular request type on the RACROUTE macro instruction. See the extended description for the system macro-request type you wish to use for specific information about the keywords available for that particular request.

specifies the address of a 512-byte work area for use by the router and the RACF front-end routine. This parameter is required for execution of the RACROUTE macro. Where it is specified can vary. For example, on MF=S it must be specified on the MF=S invocation. For MF=E, it can be specified on the MF=E invocation, on an earlier MF=M invocation that points to the same parameter list, or on the MF=L invocation that built the parameter list.

specifies whether or not REQSTOR and SUBSYS are to be used for caller identification or to determine whether RACF function is to be performed or bypassed. (See the SUBSYS keyword.)

DECOUPL=YES indicates that REQSTOR and SUBSYS parameters are used only for caller identification. Specifying DECOUPL=YES also indicates that all checking against the RACF router table is bypassed. (Checking for class entries is bypassed also.) Request processing will be performed unconditionally.

DECOUPL=NO indicates that there is checking against the RACF router table. RACF looks for a matching REQSTOR and SUBSYS parameter combination in the RACF router table; if none is found, request processing is performed just as if ACTION=RACF is specified in the router table.

DECOUPL=NO is the default.

specifies whether you want to use message return processing. You can use this parameter in conjunction with the other RACROUTE MSGxxxx parameters to store and forward messages resulting from this service.

Note: This parameter applies to REQUEST=AUTH, REQUEST=DEFINE, REQUEST=VERIFY, and REQUEST=VERIFYX.

specifies the storage subpool into which you want RACF messages returned. You can use ,MSGSP in conjunction with the other RACROUTE MSGxxxx

RACROUTE macro
parameters to store and forward RACF messages. If you do not specify a subpool, the default subpool is 0. An unauthorized caller can only specify subpools 0 through 127.

Note:
1. While IBM recommends that you specify a storage subpool, rather than allowing it to default, you should take some care in selecting a subpool, as MVS makes certain assumptions about subpool usage and characteristics. In particular, using subpool 0 or 250, or any subpool documented in z/OS MVS Programming: Assembler Services Guide as having a storage key of USER (for example, 227–231 and 241) might give unpredictable results.
2. If a common-area subpool (for example 226–228, 231, 239, 241, 245, 247, or 248) is used and not freed before the job terminates, then the job might show up in the exception reports of RMF™ (or other monitoring tools that support the tracking of common-area storage utilization) as owning common storage. Before your job terminates, it should issue a FREEMAIN to free this common storage.
3. This parameter applies to REQUEST=AUTH, REQUEST=DEFINE, REQUEST=VERIFY, and REQUEST=VERIFYX.
4. IRR008I through IRR018I, ICH408I, ICH70001I, ICH70002I, ICH70003I, ICH70004I, ICH70005I, ICH70006I, and ICH70007I messages can be returned for RACF.
5. When control returns from RACROUTE, the RACROUTE parameter-list field is mapped by SAFI MSMAD in the ICHSAFP mapping macro. SAFI MSMAD is nonzero if messages have been returned. This field contains the address of an area that consists of two fullwords followed by the message itself in write-to-operator (WTO) parameter-list format. The first word is the length of the area including the two-fullword header; the second word points to the next message area, if there is one, or contains zero if no more messages areas exist.
   You must issue the FREEMAIN macro to release the message area.

MSGSUPP=YES

specifies whether you want to suppress messages from within RACF processing. You can use this parameter in conjunction with the other RACROUTE MSGxxxx parameters to store and forward messages resulting from this service.

Note:
1. This parameter applies to REQUEST=AUTH, REQUEST=DEFINE, REQUEST=VERIFY, REQUEST=VERIFYX, REQUEST=FASTAUTH, and, for RACF only, message ICH408I and associated auditing support informational messages (IRR series), as well as ICH70001I, ICH70002I, ICH70003I, ICH70006I, and ICH70007I.
2. LOG=NONE suppresses both messages and SMF records regardless of MSGSUPP=NO.

RELATED=value

specifies information used to make notes to yourself to document macro instructions by relating functions or services to corresponding functions or services. You can use any format and put in any length and type of data you want.
RACROUTE macro

\release{number}

specifies the release level of the parameter list to be generated by this macro. Through RACF 2.2, it corresponds to the FMID of the RACF release. After that, when RACF became solely an element of OS/390® or z/OS, it corresponds to the FMID of the RACF. For example:

- 2.2 corresponds to FMID HRF2220 (RACF 2.2, OS/390 Security Server V1R1 or OS/390 Security Server V1R2)
- 2.3 corresponds to FMID HRF2230 (OS/390 Security Server V1R3)
- 2.4 corresponds to FMID HRF2240 (OS/390 Security Server V2R4)
- 2.6 corresponds to FMID HRF2260 (OS/390 Security Server V2R6)
- 2608 corresponds to FMID HRF2608 (OS/390 Security Server V2R8)
- 7703 corresponds to FMID HRF7703 (OS/390 SecureWay Security Server V2R10)
- 7705 corresponds to FMID HRF7705 (z/OS SecureWay Security Server V1R2)
- 7706 corresponds to FMID HRF7706 (z/OS Security Server V1R3)
- 7707 corresponds to FMID HRF7707 (z/OS Security Server V1R4)
- 7708 corresponds to FMID HRF7708 (z/OS Security Server V1R5)
- 7709 corresponds to FMID HRF7709 (z/OS Security Server V1R6)
- 7720 corresponds to FMID HRF7720 (z/OS Security Server V1R7)
- 7730 corresponds to FMID HRF7730 (z/OS Security Server V1R8)
- 7740 corresponds to FMID HRF7740 (z/OS Security Server V1R9)
- 7750 corresponds to FMID HRF7750 (z/OS Security Server V1R10)
- 7760 corresponds to FMID HRF7760 (z/OS Security Server V1R11)
- 7770 corresponds to FMID HRF7770 (z/OS Security Server V1R12)
- 7780 corresponds to FMID HRF7780 (z/OS Security Server V1R13)
- 7790 corresponds to FMID HRF7790 (z/OS Security Server V2R1)

To use the parameters associated with a release, you must specify the number of that release or a later release. If you specify an earlier release level, the parameter is not accepted by macro processing, and an error message is issued at assembly time. If parameters have a release dependency, the parameter descriptions with each request type identify the release number required.

The default release is 1.6.

\reqstor{reqstor addr}

specifies the address of an 8-byte character field containing the name of the piece of code that is making the request. (This address identifies a unique piece of code within a set of code that exists in a subsystem.) If this operand is omitted, a string of eight blanks is assumed.

If DECOUPL=NO is specified and you specify this operand when RACF is installed, RACF looks for a matching REQSTOR and SUBSYS parameter combination in the RACF router table; if none is found, request processing is performed just as if ACTION=RACF is specified in the router table.

For a description of the RACF router table and the macro used to update it, see “ICHFRRTB Macro” in z/OS Security Server RACF Macros and Interfaces.

\subsys{subsys addr}

specifies the address of an 8-byte character field containing the calling subsystem’s name, version, and release level. If this operand is omitted, a string of eight blanks is assumed.
If DECOUPL=NO is specified and you specify this operand when RACF is installed, RACF looks for a matching REQSTOR and SUBSYS parameter combination in the RACF router table; if none is found, request processing is performed just as if ACTION=RACF is specified in the router table.

For a description of the RACF router table and the macro used to update it, see "ICHRFRTB Macro" in z/OS Security Server RACF Macros and Interfaces.

Return codes

These return codes represent return codes from all invocations of the RACROUTE macro; for example, REQUEST=AUTH, REQUEST=VERIFY. For specific information on the success or failure of the invocation in question, see the section of this document that describes that invocation.

When you execute the macro, space for RACF return codes and their respective reason codes is reserved in the first two words of the RACROUTE parameter list. You can access them using the ICHSAFP mapping by loading the ICHSAFP pointer with the label that you specified on the list form of the macro. When control is returned, register 15 contains one of the following SAF return codes.

Note to reader

All return and reason codes are shown in hexadecimal. However, applications that call these services may convert a hex value in a register to decimal for display. Also, note that SAF return code is presented as SAF RC in the following section.

SAF RC

Meaning

00  The requested security function completed successfully. For example, if the requested function was ‘AUTH’, the authorization request was accepted.
04  The requested function has not been processed. For example, if the request was ‘AUTH’, RACF or the SAF router could neither accept nor fail the request. The following are some possible reasons for a request not to be processed.
  • The SAF router is not active.
  • The combination of class, REQSTOR, and SUBSYS was found in the RACF router table, and ACTION=None was specified.
  • The RACROUTE issuer specified DECOUPL=YES and a RELEASE= keyword with a higher release than is supported by this level of z/OS.
  • RACF is not active on the system, and RACFIND=YES was not specified, and there is no RACROUTE installation exit routine (or an exit originated a return code of 4).
  • RACF is active on the system, but no profile exists for the specified resource.
  • The class is not defined to RACF.
08  The requested function was processed by RACF, the SAF router, or the router exit (ICHRTX00) and failed. If the requested function was AUTH, the authorization request has failed. For example, if RACF is inactive for an ‘AUTH’ request with RACFIND=YES, the SAF router fails the request. The RACF- or router-exit return code and reason codes are returned in the first two words of the RACROUTE input parameter list.
**RACROUTE macro**

270F  The system could not obtain enough storage to satisfy the request. This return code may be displayed as a decimal 9999.

**Example 1**
Invoke the SAF router to perform authorization checking, using the standard form, for a non-VSAM data set residing on the volume pointed to by register 8. Register 7 points to the data set name and the RACF user is requesting the highest level of control over the data set. The “RACF-indicated” bit in the data set’s DSCB is on.

```
RACROUTE REQUEST=AUTH,WORKA=RACWK,ENTITY=((R7)),
   VOLSER=(R8),CLASS='DATASET',ATTR=ALTER,
   RACFIND=YES
```

...

```
RACWK DS CL512
```

**Example 2**
Invoke the SAF router to perform authorization checking, using the standard form, for an IMS/VS transaction pointed to by register 5. The user requests only read access.

```
RACROUTE REQUEST=FASTAUTH, WORKA=RACWK,ENTITY=(R5),
   CLASS='TIMS',WKAREA=FRACWK, ATTR=READ
```

...

```
FRACWK DS 16F
RACWK DS CL512
```

**RACROUTE (list form)**

The list form of the RACROUTE macro is written as follows. Refer to the Standard Form of the RACROUTE macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the Execute Form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
<tr>
<td>RACROUTE</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>One or more blanks must follow RACROUTE.</td>
</tr>
</tbody>
</table>

| REQUEST=type | type: System macro request type |
| WORKA=work area addr | work area addr: A-type address |
| DECOUPL=YES | Default: DECOUPL=NO |
| DECOUPL=NO |                          |
| MSGRTN=YES | Default: MSGRTN=NO |
| MSGRTN=NO |                            |
| MSGSP=subpool number | subpool number: Decimal digit 0–255; default is 0. |
| MSGSUPP=YES | Default: MSGSUPP=NO |
| MSGSUPP=NO |                             |
The REQUEST= parameters are explained under their respective invocations. The RACROUTE parameters are explained under the standard form of the RACROUTE macro with the following exception:

\texttt{\textbackslash MF=L}

specifies the list form of the RACROUTE macro instruction.

\section*{RACROUTE (execute form)}

The execute form of the RACROUTE macro is written as follows. Refer to the Standard Form of the RACROUTE macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the Execute form of the macro.

\begin{verbatim}
name name: Symbol. Begin name in column 1.

b One or more blanks must precede RACROUTE.

RACROUTE

b One or more blanks must follow RACROUTE.

REQUEST=type type: System macro request type

,WORKA=work area addr work area addr: Rx-type address or register (2) - (12)

,DECOUPL=YES,DECOUPL=NO

,MSGRTRN=YES,MSGRTRN=NO

,MSGSP=subpool number subpool number: Decimal digit 0–255

,MSGSUPP=YES,MSGSUPP=NO

,RELATED=value value: Any valid macro keyword specified

,RELEASE=number Default: RELEASE=1.6

,REQSTOR=reqstor addr regstor addr: Rx-type address or register (2) – (12)

,SUBSYS=subsys addr subsys addr: Rx-type address or register (2) – (12)
\end{verbatim}
The REQUEST= parameters are explained under their respective invocations. The RACROUTE parameters are explained under the standard form of the RACROUTE macro with the following exceptions:

\[MF=(E,ctrl\ addr)\]

specifies the execute form of the RACROUTE macro where \(ctrl\ addr\) is the address of the associated parameter list.

### RACROUTE (modify form)

The modify form of the RACROUTE macro is written as follows. Refer to the Standard Form of the RACROUTE macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the Execute form of the macro.

- **name**: Symbol. Begin \(name\) in column 1.
- b
  - One or more blanks must precede RACROUTE.
  - RACROUTE
    - b
      - One or more blanks must follow RACROUTE.

- REQUEST=\(type\)
  - \(type\): System macro request type

- \,WORKA=\(work\ area\ addr\)
  - \(work\ area\ addr\): Rx-type address or register (2) – (12)

- \,DECOUPL=\(YES\)
  - \,DECOUPL=\(NO\)

- \,MSGRTRN=\(YES\)
  - \,MSGRTRN=\(NO\)

- \,MSGP=\(subpool\ number\)
  - \(subpool\ number\): Decimal digit 0–255

- \,MSGSUPP=\(YES\)
  - \,MSGSUPP=\(NO\)

- \,RELATED=\(value\)
  - \(value\): Any valid macro keyword specified

- \,RELEASE=\(number\)
  - Default: RELEASE=1.6

- \,REQSTOR=\(reqstor\ addr\)
  - \(reqstor\ addr\): Rx-type address or register (2) – (12)

- \,SUBSYS=\(subsys\ addr\)
  - \(subsys\ addr\): Rx-type address or register (2) – (12)

- \,MF=(M,\(ctrl\ addr\))
  - \(ctrl\ addr\): Rx-type address or register (1), (2) – (12)
RACROUTE macro

The REQUEST= parameters are explained under their respective invocations. The RACROUTE parameters are explained under the standard form of the RACROUTE macro with the following exceptions:

```
,MF=(M,ctrl addr)
```

specifies the modify form of the RACROUTE macro, where `ctrl addr` is the address of the associated parameter list. The macro updates the parameter list, but does not execute the macro.
RACROUTE macro
Chapter 3. System macros

RACROUTE REQUEST=AUDIT: General-purpose security-audit request

The RACROUTE REQUEST=AUDIT macro is a general-purpose security-audit request that can be used to audit the specified resource name (ENTITYX) and action. This request records events in system-management-facilities (SMF) type 80 records, and issues messages to the network security administrator.

To use this service, you must specify RELEASE=1.9 or a later release number.

REQUEST=AUDIT is an SRB-mode-compatible service on MVS/ESA or when no profile is used (for example, in an audit initialized through LOGOPTIONS). If the caller is not in SRB mode, the caller must be APF-authorized or in supervisor state or system key.

RACROUTE REQUEST=AUDIT (standard form)

The standard form of the RACROUTE REQUEST=AUDIT macro is written as follows. For a description of additional keywords that you can code and additional parameters that are required on the RACROUTE request, but that are not specific to this request type, see "RACROUTE (standard form)" on page 13.

Note:

RACROUTE REQUEST=AUDIT requires an ACEE. For most applications, the system will have created an ACEE to represent the active user. However, for special cases where no ACEE exists, the invoker must create one before invoking RACROUTE REQUEST=AUDIT.

The most common way to create an ACEE is to issue a RACROUTE REQUEST=VERIFY, specifying ENVIR=CREATE. After all RACROUTE invocations are complete, the invoker should issue RACROUTE REQUEST=VERIFY with ENVIR=DELETE specified, to delete the ACEE previously created.

Application programs must be structured so that a task requesting RACF services does not do so while other I/O initiated by the task is outstanding. If such I/O is required, the task should either wait for the other I/O to complete before requesting RACF services, or the other I/O should be initiated under a separate task. This is necessary to assure proper processing in recovery situations.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
<tr>
<td>b</td>
<td>One or more blanks must follow RACROUTE.</td>
</tr>
</tbody>
</table>

REQUEST=AUDIT
### REQUEST=AUDIT

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>`EVENT=`event name`</td>
<td>event name: 1–8 character name</td>
</tr>
<tr>
<td>`EVENT=`event name addr`</td>
<td>event name addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>`EVQUAL=number`</td>
<td>number: 0–99</td>
</tr>
<tr>
<td>`EVQUAL=reg`</td>
<td>reg: Register (2) – (12)</td>
</tr>
<tr>
<td>`ACEE=acee addr`</td>
<td>acee addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>`CLASS=`class name`</td>
<td>class name: 1–8 character name</td>
</tr>
<tr>
<td>`CLASS=class name addr`</td>
<td>class name addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>`ENTITYX=extended resource name addr`</td>
<td>extended resource name addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>`LOGSTR=logstr addr`</td>
<td>logstr addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>RESULT=SUCCESS</td>
<td>Default: RESULT=SUCCESS</td>
</tr>
<tr>
<td>RESULT=FAILURE</td>
<td></td>
</tr>
<tr>
<td>MF=5</td>
<td></td>
</tr>
</tbody>
</table>

The parameters are explained as follows:

- **\`ACEE=acee addr\`**
  
  specifies the address of an ACEE passed on a REQUEST=AUDIT. RACF searches local profiles chained off the ACEE that have been placed there with the RACROUTE REQUEST=LIST macro.

- **\`CLASS=\`class name\`, \`CLASS=class name addr\`**
  
  specifies that you want RACF to perform authorization checking for a resource in this class. You can specify the class name or the class-name-address. If you specify a class-name-address, the address must point to an 8-byte field that contains the class name. The class name must be left-justified and padded to the right with blanks, if necessary.

  For the event “GENERAL”, REQUEST=AUDIT allows print service facility (PSF) on MVS to perform auditing. Neither profiles nor settings specified in SETROPTS LOGOPTIONS are checked. It is assumed the requester always wants an SMF record cut. If the parameters are correct, auditing is always done.

  For the class APPCLU, if SETROPTS LOGOPTIONS other than DEFAULT is specified, RACF uses the options to determine what auditing to perform. If SETROPTS LOGOPTIONS is set to DEFAULT, RACF searches the resource profile that matches the entity and uses the auditing options specified in the profile. If RACF does not find a corresponding profile, it does not perform any auditing. A message is issued to the network security administrator.

- **\`ENTITYX=extended resource name addr\`**
  
  specifies the address of a structure that consists of two 2-byte length fields, followed by the entity name.

  - The first 2-byte field specifies a buffer length that can be 0–255 bytes. This length field refers to the length of the buffer that contains the entity name; it does not include the length of either length field.
REQUEST=AUDIT

- The second 2-byte field specifies the actual length of the entity name. This length field includes the length of the actual name without any trailing blanks; it does not include the length of either length field.

These two length fields can be used in several different ways:
- If you know the length of the entity name, you can specify 0 in the first field and the length of the entity name in the second field.
- If you choose to place the entity name in a buffer area, specify the length of the buffer in the first field. For the second field, do one of the following:
  - If you know the length of the entity name, specify the length in the second field. The length of the first field can be from 0 to 255, but must be equal to or greater than the length of the second field.
  - If you do not know the length of the entity name, specify 0 in the second field, and RACF counts the number of characters in the entity name.

\texttt{EVENT=’event name’}, \texttt{EVENT=event name addr}

specifies the name of the event that you want RACF to log. You can specify the event name or the event-name address. If you specify the event-name address, it must point to an 8-byte field that contains the event name. The event name must be left-justified and padded to the right with blanks.

The events that you can log with Release 1.9 or later are APPCLU (event code 26) and GENERAL (event code 27).

The event code GENERAL allows auditing of PSF security information. PSF uses quality code 0 for this event. To achieve auditing, the \texttt{LOGSTR} and \texttt{RESULT} keywords should also be specified.

\texttt{EVQUAL=number}, \texttt{EVQUAL=reg}

specifies the event-code qualifier for the event that you want logged. If you specify a register rather than a number, you must enter the event-code qualifier in the low-order halfword of the register or the field the address in the register points to. With APPCLU, the qualifier can be from 0 to 12; with GENERAL, the qualifier can be from 0 to 99. See “SMF Records” in z/OS Security Server RACF Macros and Interfaces for a description of RACF event-code qualifiers for an event.

\texttt{LOGSTR=logstr addr}

specifies the address of a 1-byte length field followed by up to 255 bytes of character data that are written to the SMF data set together with RACF audit information. This provides a place for additional diagnostic information that the RACROUTE issuer wants included in the SMF audit records. It can be any application specific information that pertains to this particular request.

The RACF report writer puts \texttt{LOGSTR} in all its process records, but it cannot sort on the \texttt{LOGSTR} field.

\texttt{RESULT=SUCCESS}, \texttt{RESULT=FAILURE}

specifies that the resource manager (for example, PSF) can specify a RESULT keyword that causes the audit record to be marked as a success or as a failure.

The default is \texttt{RESULT=SUCCESS}.

\texttt{MF=S}

specifies the standard form of the RACROUTE REQUEST=AUDIT macro instruction.
REQUEST=AUDIT

Return codes and reason codes

When you execute the macro, space for the RACF return code and reason code is reserved in the first two words of the RACROUTE parameter list. You can access them using the ICHSAFP mapping macro by loading the ICHSAFP pointer with the label that you specified on the list form of the macro. When control is returned, register 15 contains the SAF return code.

Note

All return and reason codes are shown in hexadecimal. Also, note that SAF return code is presented as SAF RC and RACF return code is presented as RACF RC in the following section.

SAF RC

Meaning

00 RACROUTE REQUEST=AUDIT has completed successfully.

RACF RC

Meaning

00 The requested security function has completed successfully.

04 The requested function could not be performed.

Reason code

Meaning

00 RACF was not called to process the request because one of the following occurred:

• RACF is not installed.
• The combination of class, REQSTOR, and SUBSYS was found in the RACF router table, and ACTION=NONE was specified.
• The RACROUTE issuer specified DECOUPL=YES and a RELEASE= keyword with a higher release than is supported by this level of z/OS.

04 The class is not active.

08 The requested function failed.

RACF RC

Meaning

08 The class was not specified or not defined to RACF.

0C Indicates an internal error from RACXTRT.

Reason code

Meaning

xxyy xx is a return code from RACXTRT; yy is a reason code from RACXTRT.

10 Indicates parameter list-error as described by the following hex reason codes:
Reason code
Meaning
00  Event not valid
04  Event-code qualifier not valid
08  Parameter-list version not valid
0C  Parameter-list length not valid
10  Entity not valid.
14  No auditing is done. One of the following is true:
  • No profile is found and LOGOPTIONS is not set for this class.
  • No profile is found and the class RACLISTed.
  • The class is not RACLISTed.

Example 1
Invoke the RACROUTE REQUEST=AUDIT macro to search for a profile in the
APPCLU class to match the entity specified in LULUPAIR. The profiles to be
searched have been placed in storage using the RACROUTE REQUEST=LIST
macro. Be aware that if SETROPTS LOGOPTIONS other than DEFAULT has been
specified for the APPCLU class, those auditing options are the ones that RACF
uses. Set the auditing options so that an SMF 80 event APPCLU event-code
qualifier 04 (partner session keys were not equal) is logged. A message is sent to
the security console, and message ICH70005I is sent to the caller.

Note: The message cannot be received by anyone other than the caller to which it
was directed.

```
RACROUTE REQUEST=AUDIT,CLASS='APPCLU',ENTITYX=LULUPAIR, X
  ACEE=VTAMACEE,EVENT='APPCLU',EVQUAL=CODE04, X
  WORKA=RACWK,RELEASE=1.9
```

RACROUTE REQUEST=AUDIT (list form)
The list form of the RACROUTE REQUEST=AUDIT macro is written as follows.
Refer to the standard form of the RACROUTE REQUEST=AUDIT macro to
determine additional parameters that are required by the time the RACROUTE
service is invoked using the execute form of the macro.

```
<table>
<thead>
<tr>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>name: Symbol. Begin name in column 1.</td>
</tr>
</tbody>
</table>

b
One or more blanks must precede RACROUTE.

RACROUTE
b
One or more blanks must follow RACROUTE.

REQUEST=AUDIT
### REQUEST=AUDIT

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.ACEE=acee addr</td>
<td>acee addr: A-type address</td>
</tr>
<tr>
<td>.CLASS='class name'</td>
<td>class name: 1–8 character name</td>
</tr>
<tr>
<td>.CLASS=class name addr</td>
<td>class name addr: A-type address</td>
</tr>
<tr>
<td>.ENTITYX=extended resource name addr</td>
<td>extended resource name addr: A-type address</td>
</tr>
<tr>
<td>.EVENT='event name'</td>
<td>event name: 1–8 character name</td>
</tr>
<tr>
<td>.EVENT=event name addr</td>
<td>event name addr: A-type address</td>
</tr>
<tr>
<td>.EVQUAL=number</td>
<td>number: 0–99</td>
</tr>
<tr>
<td>.EVQUAL=reg</td>
<td>reg: Register (2) – (12)</td>
</tr>
<tr>
<td>.LOGSTR=logstr addr</td>
<td>logstr addr: A-type address</td>
</tr>
<tr>
<td>.RESULT=SUCCESS</td>
<td>Default: RESULT=SUCCESS</td>
</tr>
<tr>
<td>.RESULT=FAILURE</td>
<td></td>
</tr>
<tr>
<td>.MF=L</td>
<td></td>
</tr>
</tbody>
</table>

The parameters are explained under the standard form of the RACROUTE REQUEST=AUDIT macro with the following exception:

`.MF=L`

specifies the list form of the RACROUTE REQUEST=AUDIT macro instruction.

### RACROUTE REQUEST=AUDIT (execute form)

The execute form of the RACROUTE REQUEST=AUDIT macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=AUDIT macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
</tbody>
</table>

RACROUTE

| b    | One or more blanks must follow RACROUTE. |

REQUEST=AUDIT

| .ACEE=acee addr | acee addr: Rx-type address or register (2) – (12) |
| .CLASS=class name addr | class name addr: Rx-type address or register (2) – (12) |
| .ENTITYX=extended resource name addr | extended resource name addr: Rx-type address or register (2) – (12) |
| .EVENT=event name addr | event name addr: Rx-type address or register (2) – (12) |
| .EVQUAL=number | number: 0–99 |
| .EVQUAL=reg | reg: Register (2) – (12) |
| .LOGSTR=logstr addr | logstr addr: Rx-type address or register (2) – (12) |
REQUEST=AUDIT

,RESULT=SUCCESS
,RESULT=FAILURE

,MF=(E,ctrl addr)  
ctrl addr: Rx-type address or register (1) – (12)

The parameters are explained under the standard form of the RACROUTE REQUEST=AUDIT macro with the following exception:

,MF=(E,ctrl addr)  
specifies the execute form of the RACROUTE REQUEST=AUDIT macro instruction.

RACROUTE REQUEST=AUDIT (modify form)

The modify form of the RACROUTE REQUEST=AUDIT macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=AUDIT macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

name

name: Symbol. Begin name in column 1.

b

One or more blanks must precede RACROUTE.

RACROUTE

b

One or more blanks must follow RACROUTE.

REQUEST=AUDIT

,ACEE=acee addr  
acee addr: Rx-type address or register (2) – (12)

,CLASS=class name addr  
class name addr: Rx-type address or register (2) – (12)

,ENTITYX=extended resource name addr  
extended resource name addr: Rx-type address or register (2) – (12)

,EVENT=event name addr  
event name addr: Rx-type address or register (2) – (12)

,EVQUAL=number

number: 0–99

,EVQUAL=reg

reg: Register (2) – (12)

,LOGSTR=logstr addr  
logstr addr: Rx-type address or register (2) – (12)

,RESULT=SUCCESS
,RESULT=FAILURE

,MF=M

The parameters are explained under the standard form of the RACROUTE REQUEST=AUDIT macro with the following exception:
REQUEST=AUDIT

\texttt{,MF=M}

specifies the modify form of the RACROUTE REQUEST=AUDIT macro instruction.

---

**RACROUTE REQUEST=AUTH: Check RACF authorization**

The RACROUTE REQUEST=AUTH macro checks a user's authority to access a resource, based on a profile in the RACF database when a user requests access to a RACF-protected resource.

The types of checks available are:

1. **First-party**: You do not provide any of the applicable parameters and RACF locates your ACEE and checks your authority.
2. **Second-party**: You provide only an ACEE (without the UTOKEN, USERID, and GROUPID parameters), and RACF checks that user's authority instead of yours.
3. **Third-party**: You provide UTOKEN, USERID, or GROUPID and RACF checks the specified user's or group's authority. This requires use of REQUEST=VERIFY by the REQUEST=AUTH processing, and AUTH anchors the resulting ACEE in your ACEE (or in the ACEE specified by ACEE). This circumvents the VERIFY processing if another check for the same user is required later on.

For third-party authorization checking, RACF performs the following steps:

1. If the USERID keyword is *NONE* and GROUPID is not specified, RACF checks using a default (undefined-user) ACEE.
2. Before building the security environment (ACEE) to perform this third-party authorization request, RACF checks to see if an ACEE is already available that is compatible with the USERID, GROUPID, and/or UTOKEN specified on this RACROUTE request. The ACEE can be either:
   - The ACEE of the user issuing the RACROUTE
   - The most recent ACEE created by that user when doing a previous third-party authorization request.

   If either of these ACEEs is compatible with the current RACROUTE request, RACF uses the existing ACEE rather than creating a new one.
3. The user specified by USERID or UTOKEN must not be a revoked user, or the RACROUTE request fails.

**Note:**

1. If the calling program does not specify the GROUPID keyword, the internal RACROUTE REQUEST=VERIFY function uses the default group associated with the specified user ID.
2. If an ACEE with a user ID of *BYPASS* is used, a return code of 4 is returned. Depending upon the application, this may allow access to a resource.

The RACROUTE REQUEST=AUTH caller must be authorized (APF-authorized in system key 0–7, or in supervisor state) for certain keywords. See the keyword descriptions for authorization requirements.

The caller cannot hold any locks when issuing RACROUTE REQUEST=AUTH.
RACROUTE REQUEST=AUTH (standard form)

The standard form of the RACROUTE REQUEST=AUTH macro is written as follows. For a description of additional keywords that you can code and additional parameters that are required on the RACROUTE request, but that are not specific to this request type, see “RACROUTE (standard form)” on page 13.

Note:

RACROUTE REQUEST=AUTH requires an ACEE. For most applications, the system will have created an ACEE to represent the active user. However, for special cases where no ACEE exists, the invoker must create one before invoking RACROUTE REQUEST=AUTH. If there is no ACEE, the result is ABEND 582 with reason code 00.

The most common way to create an ACEE is to issue a RACROUTE REQUEST=VERIFY, specifying ENVIR=CREATE. After all RACROUTE invocations are complete, the invoker should issue RACROUTE REQUEST=VERIFY with ENVIR=DELETE specified, to delete the ACEE previously created.

Application programs must be structured so that a task requesting RACF services does not do so while other I/O initiated by the task is outstanding. If such I/O is required, the task should either wait for the other I/O to complete before requesting RACF services, or the other I/O should be initiated under a separate task. This is necessary to assure proper processing in recovery situations.

name

name: Symbol. Begin name in column 1.

b

One or more blanks must precede RACROUTE.

RACROUTE

b

One or more blanks must follow RACROUTE.

REQUEST=AUTH

,CLASS=’class name’
class name: 1–8 character name
,CLASS=class name addr
class name addr: A-type address or register (2) – (12)
,ENTITY=resource name addr
resource name addr: A-type address only
,ENTITY=resource name addr)
resource name addr: A-type address or register (2) – (12)
,ENTITY=resource name addr,CSA
,ENTITY=resource name addr,PRIVATE
,ENTITY=resource name addr,NONE
,ENTITYX=extended resource name addr
extended resource name addr: A-type address only
,ENTITYX=(extended resource name addr)
extended resource name addr: A-type address or register (2) – (12)
,ENTITYX=(extended resource name addr,CSA)
,ENTITYX=(extended resource name addr,PRIVATE)
,ENTITYX=(extended resource name addr,NONE)
**REQUEST=AUTH**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>.PROFILE=profile addr</code></td>
<td><code>profile addr</code>: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td><code>.VOLSER=vol addr</code></td>
<td><code>vol addr</code>: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td><strong>Note:</strong> VOLSER is required for CLASS=DATASET and DSTYPE not equal to M when a discrete profile name is used and when ENTITY is also coded.</td>
<td></td>
</tr>
<tr>
<td><code>.ACCLVL=access level addr</code></td>
<td><code>access level addr</code>: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td><code>.ACCLVL=(access level addr, parm list addr)</code></td>
<td><code>parm list addr</code>: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td><code>.ACEE=acee addr</code></td>
<td><code>acee addr</code>: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td><code>.APPL='applname'</code></td>
<td><code>applname</code>: 1–8 character name</td>
</tr>
<tr>
<td><code>.APPL=applname addr</code></td>
<td><code>applname addr</code>: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td><code>.ATTR=READ</code></td>
<td><strong>Default:</strong> ATTR=READ</td>
</tr>
<tr>
<td><code>.ATTR=UPDATE</code></td>
<td></td>
</tr>
<tr>
<td><code>.ATTR=CONTROL</code></td>
<td></td>
</tr>
<tr>
<td><code>.ATTR=ALTER</code></td>
<td></td>
</tr>
<tr>
<td><code>.ATTR=reg</code></td>
<td><code>reg</code>: register (2) – (12)</td>
</tr>
<tr>
<td><code>.DSTYPE=N</code></td>
<td><strong>Default:</strong> DSTYPE=N</td>
</tr>
<tr>
<td><code>.DSTYPE=V</code></td>
<td></td>
</tr>
<tr>
<td><code>.DSTYPE=M</code></td>
<td></td>
</tr>
<tr>
<td><code>.DSTYPE=T</code></td>
<td></td>
</tr>
<tr>
<td><code>.FILESEQ=number</code></td>
<td><code>number</code>: 1–65535</td>
</tr>
<tr>
<td><code>.FILESEQ=reg</code></td>
<td><code>reg</code>: register (2) – (12)</td>
</tr>
<tr>
<td><code>.GENERIC=YES</code></td>
<td><strong>Default:</strong> GENERIC=ASIS</td>
</tr>
<tr>
<td><code>.GENERIC=ASIS</code></td>
<td></td>
</tr>
<tr>
<td><code>.GROUPID='groupid'</code></td>
<td><code>groupid</code>: 1–8 character group ID</td>
</tr>
<tr>
<td><code>.GROUPID=groupname addr</code></td>
<td><code>groupname addr</code>: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td><code>.INSTLN=parm list addr</code></td>
<td><code>parm list addr</code>: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td><code>.LOG=ASIS</code></td>
<td><strong>Default:</strong> LOG=ASIS</td>
</tr>
<tr>
<td><code>.LOG=NOFAIL</code></td>
<td></td>
</tr>
<tr>
<td><code>.LOG=NONE</code></td>
<td></td>
</tr>
<tr>
<td><code>.LOG=NOSTAT</code></td>
<td></td>
</tr>
<tr>
<td><code>.LOGSTR=logstr addr</code></td>
<td><code>logstr addr</code>: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td><code>.OLDVOL=old vol addr</code></td>
<td><code>old vol addr</code>: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td><code>.RACFIND=YES</code></td>
<td></td>
</tr>
<tr>
<td><code>.RACFIND=NO</code></td>
<td></td>
</tr>
<tr>
<td><code>.RECVR=recvr addr</code></td>
<td><code>recvr addr</code>: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td><code>.RTOKEN=rtoken addr</code></td>
<td><code>rtoken addr</code>: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td><code>.STATUS=NONE</code></td>
<td><strong>Default:</strong> STATUS=NONE</td>
</tr>
<tr>
<td><code>.STATUS=ERASE</code></td>
<td></td>
</tr>
<tr>
<td><code>.STATUS=EVERDOM</code></td>
<td></td>
</tr>
<tr>
<td><code>.STATUS=WRITEONLY</code></td>
<td></td>
</tr>
</tbody>
</table>
REQUEST=AUTH

.STATUS=ACCESS

.SYSTEM=NO
SYSTEM=YES

.TAPELBL=STD
.TAPELBL=BLP
.TAPELBL=NL

.USERID='userid'
USERID=userid

.U_TOKEN=token addr

.MF=S

When SYSTEM=YES is specified, only the following keywords are valid:

.CLASS='class name'
CLASS=class name

.ENTITY=resource name addr
ENTITY=(resource name addr)

.ATTR=READ
ATTR=UPDATE
ATTR=CONTROL
ATTR=ALTER
 ATTR=reg

.APPL='applname'
APPL=applname

.INSTLN=parm list addr

.LOG=ASIS
LOG=NOFAIL
LOG=NONE
LOG=NOSTAT

The parameters are explained as follows:

.ACCLVL=access level addr
.ACCLVL=(access level addr,parm list addr)
 specifies the tape-label access-level information for the MVS tape-label functions. The access level pointed to by the specified address is a 1-byte length field, containing the value (0–8) of the length of the following data, followed by an 8-character string that is passed to the RACROUTE REQUEST=AUTH installation-exit routines. The optional parameter list pointed to by the specified address contains additional information to be passed to the RACROUTE REQUEST=AUTH installation exit routines. RACF does not inspect or modify this information.

.ACCE=acee addr
 specifies the address of the ACEE to be used during RACF authorization-check processing.
REQUEST=AUTH

Programs must be APF-authorized, system key 0-7, or in supervisor state to use the ACEE parameter.

If no ACEE is specified, RACF uses the TASK ACEE pointer (TCBSENV) in the extended task control block (TCB). Otherwise, or if the TASK ACEE pointer is zero, RACF uses the main ACEE for the address space. The main ACEE is pointed to by the ASXBSENV field of the address-space extension block.

,APPL='applname'
,APPL=applname addr
specifies the name of the application requesting authorization checking. The application name is not used for the authorization checking process but is made available to the installation exit routine or routines called by the RACROUTE REQUEST=AUTH routine. If the address is specified, the address must point to an 8-byte field containing the application name, left-justified and padded with blanks.

,ATTR=READ
,ATTR=UPDATE
,ATTR=CONTROL
,ATTR=ALTER
,ATTR=reg
specifies the level of authority requested. RACF checks the resource profile protecting the resource identified by the ENTITY and CLASS keywords. The values have the following hierarchical order:

• READ
• UPDATE
• CONTROL
• ALTER

That is, if a user has update authority and ATTR=READ is specified, RACF returns a return code of 0. If ATTR=CONTROL, RACF returns a return code of 8.

For multilevel secure environments:
1. When ATTR=READ or ALTER, it will be treated as though it was a read-only request for purposes of mandatory access control (MAC) checking.
2. When ATTR=UPDATE or CONTROL, it will be treated as though it was a read-write request for purposes of mandatory access control (MAC) checking.

If a register is specified, the register must contain one of the following codes in the low-order byte of the register:

X'02'  READ
X'04'  UPDATE
X'08'  CONTROL
X'80'  ALTER

The default is ATTR=READ.

,CLASS='class name'
,CLASS=class name addr
specifies that RACF authorization checking is to be performed for a resource of the specified class. The address must point to a 1-byte field indicating the length of the class name, followed by the class name.
REQUEST=AUTH

The specified class must be defined in the class descriptor table, and must be active for this request to be processed. In addition, if the class descriptor table specifies that RACLST is required, the SETROPTS RACLST option must be active for the class.

\,DSTYPE=N
\,DSTYPE=V
\,DSTYPE=M
\,DSTYPE=T

specifies the type of data set associated with the request:

N for non-VSAM
V for VSAM
M for model profile
T for tape

DSTYPE=T should not be specified unless the SETROPTS TAPEDSN option is active (RCVTTSN bit is on); otherwise, the processing is the same as for RACROUTE REQUEST=AUTH, CLASS='TAPEVOL'.

DSTYPE should be specified only for CLASS=DATASET.

\,ENTITY=resource name addr
\,ENTITY=(resource name addr)
\,ENTITY=(resource name addr,CSA)
\,ENTITY=(resource name addr,PRIVATE)
\,ENTITY=(resource name addr,NONE)
\,ENTITYX=extended resource name addr
\,ENTITYX=(extended resource name addr)
\,ENTITYX=(extended resource name addr,CSA)
\,ENTITYX=(extended resource name addr,PRIVATE)
\,ENTITYX=(extended resource name addr,NONE)
\,PROFILE=profile addr

specifies the resource address.

**Guideline:** Use ENTITYX rather than ENTITY. With ENTITY, the entity name you pass to RACF must be in a buffer, the size of which is determined by the length in the class descriptor table. If the maximum length of a class descriptor entity increases in the future, you must modify your program to use a larger buffer. By using ENTITYX, you avoid this possible problem because you remove the class descriptor table dependency from your program.

For the ENTITY keyword, the resource name is a 44-byte DASD data set name for CLASS=DATASET, or a 6-byte volume serial number for CLASS=DASDVOL or CLASS=TAPEVOL. The length of all other resource names is determined from the class descriptor table.

- ENTITY=resource name addr or ENTITY=(resource name addr) specifies that RACF authorization checking is to be performed for the resource whose name is pointed to by the specified address. The name must be left-justified in the field and padded with blanks.
- ENTITY=(resource name addr,CSA) specifies that RACF authorization checking is to be performed for the indicated resource and that a copy of the profile is to be maintained in central storage. The storage acquired for the profile is obtained from the common storage area (CSA), and is fetch-protected, key 0 storage.

For ENTITY, CSA returns the address of the in-storage profile, with the name field replaced by the entity name specified.
If CSA is specified and the return code from the RACROUTE REQUEST=AUTH macro instruction is 00 or 08 (that is, a profile exists), the address of the profile mapped by ICHRRPF is returned in register 1, as long as the return code is not a CDT default return code for a resource profile. If a default return code is returned, register 1 does not contain the address of the profile. Note that, like CSA, when PRIVATE is specified, the profile is not returned along with a default return code. See note 3 for the effect of a default return code on a reason code.

Programs must be APF-authorized, system key 0–7, or in supervisor state to specify CSA with the ENTITY keyword.

Note:

1. If a common-area subpool (for example 226–228, 231, 239, 241, 245, 247, or 248) is used and not freed before the job terminates, then the job might show up in the exception reports of RMF (or other monitoring tools that support the tracking of common-area storage utilization) as owning common storage. Before your job terminates, it should issue a FREEMAIN to free this common storage.

2. If a VOLSER is specified on the RACROUTE REQUEST=AUTH macro for the DATASET class, it is built into the profile.

3. When a default return code of other than 4 is specified for a class in the class descriptor table, in addition to returning that specified return code, the reason code is incremented by X’200’ (decimal 512).

- ENTITY=(resource name addr, PRIVATE) PRIVATE specifies the same as CSA except that RACROUTE returns the profile in the user private area rather than in common storage, and the name field contains the name of the returned profile instead of the name of the resource that was specified on the ENTITY keyword. The issuer of RACROUTE REQUEST=AUTH must free this storage when the profile is no longer needed. (The profile subpool number and length are returned as well as the profile data.) For default reason codes, no profile is returned as for CSA.

If the reason codes default, refer to “Class descriptor table (CDT) default return codes and reason codes” on page 47 to identify them. If CSA or PRIVATE was specified on ENTITY or ENTITYX, register 1 does not point to a profile.

Programs must be APF-authorized, system key 0–7, or in supervisor state to specify PRIVATE with the ENTITY keyword.

Note: If a VOLSER is specified on the RACROUTE REQUEST=AUTH macro for the DATASET class, it is built into the profile.

- ENTITY=(resource name addr, NONE) specifies the same as ENTITY=resource-name address. However, no profile is returned.

- ENTITYX=extended resource address or ENTITYX=(extended resource address) specifies the address of a structure that consists of two 2-byte length fields, followed by the entity name. The entity name is the name of the resource for which RACF authorization checking is to be performed.
  - The first 2-byte field specifies a buffer length, which can be from 0 to 255 bytes. This length field refers to the length of the buffer that contains the entity name; it does not include the length of either length field.
  - The second 2-byte field specifies the actual length of the entity name. This length field includes the length of the actual name without any trailing blanks; it does not include the length of either length field.

These two length fields can be used in several different ways:
If you know the length of the entity name, you can specify 0 in the first field and the length of the entity name in the second field.

If you choose to place the entity name in a buffer area, specify the length of the buffer in the first field. For the second field, do one of the following:

- If you know the length of the entity name, specify the length in the second field. The length of the first field can be from 0 to 255, but must be equal to or greater than the length of the second field.
- If you do not know the length of the entity name, specify 0 in the second field, and RACF counts the number of characters in the entity name.

To use this keyword, you must also specify RELEASE=1.9 or later.

**ENTITYX=(extended resource name addr,CSA)** specifies that RACF authorization checking is to be performed for the indicated resource, and that a copy of the profile is to be maintained in main storage. The storage acquired for the profile is obtained from the common storage area (CSA), and is fetch-protected, key 0 storage.

For ENTITYX, CSA returns the address of the in-storage profile, with the name field replaced by the entity name specified.

If CSA is specified and the return code produced by the RACROUTE REQUEST=AUTH macro instruction is 00 or 08, the address of the profile mapped by ICHRRPF is returned in register 1, as long as the return code is not a CDT default return code for a resource profile. If a default return code is returned, register 1 does not contain the address of the profile. Note that, like CSA, when PRIVATE is specified, the profile is not returned along with a default return code. See note 2 for the effect of a default return code on a reason code.

Programs must be APF-authorized, system key 0–7, or in supervisor state to specify CSA with the ENTITYX keyword.

**Note:**

1. When a common-area subpool (for example 226–228, 231, 239, 241, 245, 247, or 248) is used and not freed before the job terminates, then the job might show up in the exception reports of RMF (or other monitoring tools that support the tracking of common-area storage utilization) as owning common storage. Before your job terminates, it should issue a FREEMAIN to free this common storage.

2. When a default return code of other than 4 is specified for a class in the class descriptor table, in addition to returning that specified return code, the reason code is incremented by 'X'200' (decimal 512).

To use this keyword, you must also specify RELEASE=1.9 or a later release number.

**ENTITYX=(extended resource name addr,PRIVATE)** PRIVATE specifies the same as CSA, except that RACROUTE returns the profile in the user private area rather than in common storage, and the name field contains the name of the returned profile instead of the name of the resource that was specified on the ENTITY keyword. For default reason codes, no profile is returned as for CSA.

If the reason codes default, refer to "Class descriptor table (CDT) default return codes and reason codes" on page 47 to identify them. If CSA or PRIVATE was specified on ENTITY or ENTITYX, then register 1 does not point to a profile.
The issuer of RACROUTE REQUEST=AUTH must free this storage when the profile is no longer needed. (The profile subpool number and length are part of the profile data returned.)

To use this keyword, you must also specify RELEASE=1.9 or a later release number.

Programs must be APF-authorized, system key 0–7, or in supervisor state to specify PRIVATE with the ENTITYX keyword.

• ENTITYX=(extended resource name addr,NONE) specifies the same as ENTITYX=resource name address. However, no profile is returned.

To use this keyword, you must also specify RELEASE=1.9 or a later release number.

• PROFILE=profile addr specifies that RACF authorization checking is to be performed for the resource whose profile is pointed to by the specified address. This profile must be supplied by a previously executed RACROUTE REQUEST=AUTH with CSA or PRIVATE specified. A profile supplied by RACROUTE REQUEST=LIST is not acceptable.

To specify PROFILE, programs must be APF-authorized and in supervisor state. The programs must also be in system key 0 or in the same key as the storage of the profile.

Note:

1. If ENTITY=(...,CSA) or ENTITY=(...,PRIVATE) is coded on the RACROUTE macro instruction, RACF ignores the privileged and trusted attributes and performs normal authorization processing.

2. It is generally not advisable to use generic characters in an entity(x) name as the results you expect usually will not occur. Using a generic character in the entity(x) name indicates that the application is (for some reason) asking about a profile, not a resource, and therefore during processing of the AUTH request RACF will use only a generic profile name that is exactly the same in the RACF data base. For example: An entity name of A.B.C*.D will ONLY match on a generic profile of that exact name in that resource class. Less specific (such as A.B.**) generic profiles will NOT be used.

FILESEQ=number

specifies the file-sequence number of a tape data set on a tape volume or within a tape-volume set. The number must be in the range 1–65535. If a register is specified, it must contain the file-sequence number in the low-order halfword. If CLASS=DATASET and DSTYPE=T are not specified, FILESEQ is ignored.

GENERIC=YES
GENERIC=ASIS

specifies whether the resource name is to be treated as a generic profile name. GENERIC is ignored if the GENCMD option on the RACF SETROPTS command is not specified for the class (see z/OS Security Server RACF Command Language Reference).

This keyword is designed primarily for use by RACF commands.

YES The resource name is considered a generic profile name, even if it does not contain a generic character: an asterisk (*), a percent sign (%), or, for general-resource classes, an ampersand sign (&). If you specify
REQUEST=AUTH

GENERIC=YES, the entity(x) name in the macro will match only a
generic profile name that is exactly the same in the RACF data base. It
will not match a discrete name.

ASIS The resource name is considered generic if it contains a generic
character: an asterisk (*), a percent sign (%), or, for general-resource
classes, an ampersand sign (&).

\GROUPID='groupid'
\GROUPID=groupid address
specifies the group name that RACF uses to perform third-party authorization
checking. This is an 8-character field, left-justified, and padded to the right
with blanks.

If the calling program wants a third-party authorization check performed on
the group name rather than the user ID, the USERID keyword must be
specified as *NONE*. That is, when the caller invokes third-party authorization
checking, RACF verifies the authority of the group name to the requested
resource; RACF disregards the group name associated with the ACEE of the
caller.

Programs must be APF-authorized, system key 0–7, or in supervisor state to
use the GROUPID keyword.

\INSTLN=parm list addr
specifies the address of an area that is to contain parameter information
meaningful to the RACROUTE REQUEST=AUTH installation exit routine. This
information is passed to the installation exit routine when it is given control by
RACROUTE REQUEST=AUTH.

The INSTLN parameter can be used by an application program acting as a
resource manager that needs to pass information to the RACROUTE
REQUEST=AUTH installation exit routine.

\LOG=ASIS
\LOG=NOFAIL
\LOG=NONE
\LOG=NOSTAT
specifies the types of access attempts to be recorded on the SMF data set.

ASIS RACF records the event in the manner specified in the profile that
protects the resource, or by other methods such as a SETROPTS option.

NOFAIL
If the authorization check fails, the attempt is not recorded. If the
authorization check succeeds, the attempt is recorded as in ASIS.

Note: When SETROPTS PROTECTALL(WARNING) is in effect, the
attempt is recorded as for ASIS.

NONE
The attempt is not recorded.

LOG=NONE suppresses both messages and SMF records regardless of
MSGSUPP=NO and MSGRTRN.

NOSTAT
Like LOG=NONE, the attempt is not recorded and it suppresses both
messages and SMF records regardless of the MSGSUPP and MSGRTRN
keyword values. It differs in that, even if resource statistic gathering
had been requested, it would not occur.
REQUEST=AUTH

Programs must be APF-authorized, system key 0–7, or in supervisor state to use the NOFAIL, NONE, and NOSTAT keywords.

,LOGSTR=logstr addr
specifies a variable-length data string consisting of a 1-byte, binary length field followed by character data that is to be included in the RACF SMF process records. The character data can be 0–255 bytes long. The RACF report writer includes LOGSTR data on the process reports.

To use this keyword, you must also specify RELEASE=1.9 or a later release number.

,OLDVOL=old vol addr
specifies a volume serial number:
• For CLASS=DATASET, within the same multivolume data set specified by VOLSER=
• For CLASS=TAPEVOL, within the same tape volume specified by ENTITY=.

RACF authorization checking verifies that the OLDVOL specified is part of the same multivolume data set or tape-volume set. RACF authorization checking does not look at global access table entries when the OLDVOL parameter is specified.

The specified address points to the field that contains the volume serial number padded to the right with blanks, if necessary, to make 6 characters.

,RACFIND=YES
,RACFIND=NO
indicates whether the resource is meant to be protected by a discrete profile. The RACF processing and the possible return codes are given in Table 3.

Note: In all cases, a return code of X'0C' is also possible if the OLDVOL specified was not part of the multivolume data set defined by VOLSER, or if it was not part of the same tape volume defined by ENTITY.

Table 3. Types of profile checking performed by RACROUTE REQUEST=AUTH

<table>
<thead>
<tr>
<th>Operand</th>
<th>Generic profile checking inactive</th>
<th>Generic profile checking active</th>
</tr>
</thead>
<tbody>
<tr>
<td>RACFIND=YES</td>
<td>Look for discrete profile; if found, exit with return code 00 or 08. If no discrete profile is found, exit with return code 08.</td>
<td>Look for discrete profile; if found, exit with return code 00 or 08. Look for generic profile; if found, exit with return code 00 or 08. Exit with return code 08 if neither a discrete nor a generic profile is found.</td>
</tr>
<tr>
<td>RACFIND=NO</td>
<td>No checking. Exit with return code 04. (See note).</td>
<td>Look for generic profile; if found, exit with return code 00 or 08. If not found, exit with return code 04. (See note).</td>
</tr>
<tr>
<td>RACFIND not specified</td>
<td>Look for discrete profile; if found, exit with return code 00 or 08. If no discrete profile is found, exit with return code 04. (See note).</td>
<td>Look for discrete profile; if found, exit with return code 00 or 08. Look for generic profile; if found, exit with return code 00 or 08. Exit with return code 04 if neither a discrete nor a generic profile is found. (See note).</td>
</tr>
</tbody>
</table>

Note: If PROTECTALL is active, no profile is found, and the user ID whose authority was checked does not have the SPECIAL attribute, RACF returns a return code X'08' instead of a return code X'04' and denies access.
REQUEST=AUTH

,RECVR=recvr addr
specifies the address of the user ID that has the authority to access the resource regardless of whether there is a resource profile to protect it. The field is 8 bytes, left-justified and padded to the right with blanks.

The RTOKEN= keyword is required when the RECVR= keyword is specified.

To use this keyword, you must also specify RELEASE=1.9 or a later release number.

,RTOKEN=rtoken addr
specifies the address of the RTOKEN of a unit of work. See the explanation of UTOKEN for format.

To use this keyword, you must also specify RELEASE=1.9 or later.

,STATUS=None
,STATUS=ERASE
,STATUS=EVERDOM
,STATUS=WRITEONLY
,STATUS=ACCESS
specifies the type of status required.

NONE
No STATUS= functions have been requested.

ERASE
RACROUTE REQUEST=AUTH is to return the ERASE status of the data set specified on the ENTITY or ENTITYX keyword. The ERASE status is returned as the RACF reason code. A reason code of 4 indicates the data set will be erased when scratched, and a reason code of 0 indicates it will not. The user of this operand should be aware that the SETROPTS ERASE setting, in conjunction with the ERASE setting in the profile protecting the data set, determines the ERASE status. This parameter is valid for CLASS=DATASET and a DSTYPE value other than T.

EVERDOM
Security-label authorization checking includes a check to see whether the user has a security label, other than that of this job or logon session, that could ever dominate that of the current object. This is done primarily so that message security can determine what to do with the messages that cannot currently be shown to the user. For example, if the user does not have a security label that can ever dominate that of the message, the message can be deleted. Be aware that choosing this option increases processing time. The default is that security-label authorization checking occurs with the security label of the current job or logon session.

STATUS=EVERDOM is intended for use only with ATTR=READ and is supported for classes using the following authorization checking only:
1. EQUALMAC
2. MAC
3. RVRSMAC

WRITEONLY
The request is for output only in a class that also allows read or write functions. No reading is to be done.
REQUEST=AUTH

STATUS=WRITEONLY is intended for use by a trusted process that only allows its users to write data, not to read it, and is supported for classes using the following authorization checking only:
1. EQUALMAC
2. MAC

ACCESS
The request is simply to return the user’s highest current access to the resource specified. Upon successful completion, the user’s access is returned in the RACF reason code. No auditing is done for this request.

Note:
1. If the ATTR= keyword is specified along with STATUS=ACCESS, the ATTR= keyword is ignored.
2. To use the STATUS=ACCESS keyword, you must specify RELEASE=1.9 or later.

,SYSTEM=NO
,SYSTEM=YES
specifies whether the caller is in system key 0–7 or supervisor state, or both.

NO indicates that the caller cannot guarantee to be in supervisor state or system key 0–7. When SYSTEM=NO is specified, normal REQUEST=AUTH processing occurs.

YES indicates that the caller is in system key 0–7 or supervisor state, or both. If the caller is not in system key 0–7 or supervisor state, an abend might occur. Specifying SYSTEM=YES when the caller is in system key 0–7 or supervisor state might allow more efficient processing of this request. Currently, SYSTEM=YES has no effect on RACF’s processing of RACROUTE REQUEST=AUTH.

Note: To use the SYSTEM= keyword you must specify RELEASE=1.9.2 or later.
The default is SYSTEM=NO.

,TAPELBL=STD
,TAPELBL=BLP
,TAPELBL=NL
specifies the type of tape-label processing to be done:

STD IBM or ANSI standard labels
BLP Bypass label processing
NL Non-labeled tapes

For TAPELBL=BLP, the user must have the requested authority to the profile ICHBLP in the general-resource class FACILITY. For more information about using the ICHBLP profile, see z/OS Security Server RACF Security Administrator’s Guide

For TAPELBL=NL or BLP, data management routines do not allow the user to protect volumes with volume serial number in the format "Lnnnnn."

This parameter is primarily intended for use by data-management routines to indicate the label type from the LABEL keyword on the JCL statement.
This parameter is valid for CLASS=DATASET and DSTYPE=T, or CLASS=TAPEVOL.

\texttt{USERID='userid'}
\texttt{USERID=userid address}

specifies the user ID that RACF uses to perform third-party authorization checking. This is an 8-character field that is left-justified and padded to the right with blanks.

If USERID is specified when the caller invokes RACROUTE REQUEST=AUTH, RACF verifies that user's authority to the given entity; RACF disregards the user ID associated with the ACEE of the caller.

\textbf{Note:}
1. If the calling program does not specify the GROUPID keyword, the internal RACROUTE REQUEST=VERIFY function uses the default group associated with the specified user ID.
2. Specifying USERID=BLANKS (where BLANKS is eight characters of X'40' characters), with GROUPID not specified or specified as GROUPID=BLANKS, causes RACF to build an ACEE with an asterisk (*) specified as the user ID or group name. This is the same as an ACEE built by RACROUTE REQUEST=VERIFY without specifying USERID, GROUPID, or PASSWORD.
3. An ACEE with a user ID of asterisk (*) is built by RACF only for the case listed in Note 2, and it cannot be specified. For example, consider USERID=VAR1, where variable VAR1 is the asterisk character (*) followed by 7 blanks. This results in a 282-48 abend that regards the asterisk (*) as in invalid RACF user ID.

Programs must be APF-authorized, system key 0–7, or in supervisor state to use the USERID keyword.

\texttt{UTOKEN=token addr}

specifies the address of the UTOKEN of the user for whom RACF performs third-party authorization checking. The first byte contains the length of the UTOKEN, and the second byte contains the version number.

If UTOKEN is specified when the caller invokes RACROUTE REQUEST=AUTH, RACF verifies that user's authority to the given entity; RACF disregards the user ID associated with the ACEE of the caller. Furthermore, if this parameter is specified, it takes precedence over the USERID and GROUPID parameters (if specified).

To use this keyword, you must also specify RELEASE=1.9 or a later release number.

The ACEE= does not perform a third-party check. Only UTOKEN, USERID, and GROUPID do this.

Programs must be APF-authorized, system key 0–7, or in supervisor state to use the UTOKEN parameter.

\texttt{VOLSER=vol addr}

specifies the volume serial number, as follows:
\begin{itemize}
  \item For non-VSAM DASD data sets and tape data sets, this is the volume serial number of the volume on which the data set resides.
  \item For VSAM DASD data sets, this is the volume serial number of the catalog controlling the data set.
\end{itemize}
REQUEST=AUTH

The volume serial number is optional if DSTYPE=M is specified; it is ignored if the profile name is generic.

The field pointed to by the specified address contains the volume serial number, padded to the right with blanks if necessary to make six characters. VOLSER= is only valid (and must be supplied) with CLASS=DATASET, (unless DSTYPE=M is specified) when ENTITY or ENTITYX is also coded.

\$MF=S

specifies the standard form of the RACROUTE REQUEST=AUTH macro instruction.

Return codes and reason codes

When you execute the macro, space for the RACF return code and reason code is reserved in the first two words of the RACROUTE parameter list. You can access them using the ICHSAFP mapping macro by loading the ICHSAFP pointer with the label that you specified on the list form of the macro. When control is returned, register 15 contains the SAF return code.

<table>
<thead>
<tr>
<th>SAF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACROUTE REQUEST=AUTH completed successfully.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>The user is authorized by RACF to obtain use of a RACF-protected resource.</td>
</tr>
</tbody>
</table>

**Reason code**

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
</tr>
<tr>
<td>04</td>
</tr>
</tbody>
</table>

- STATUS=ERASE was specified and the data set is to be erased when scratched, or
- The warning status of the resource was requested by the RACROUTE REQUEST=AUTH issuer's setting bit X'10' at offset 12 decimal in the request-specific portion of the RACROUTE REQUEST=AUTH parameter list, and authorization was granted because WARNING was specified in the profile protecting the resource. The X'10' at offset 12 bit is not a programming interface. The request-specific portion of the RACROUTE REQUEST=AUTH parameter list follows the RACROUTE parameter list (ICHSAFP) and is mapped by the mapping macro, ICHACHKL.

| 10 | When CLASS=TAPEVOL, indicates the TAPEVOL profile contains a TVTOC. |
REQUEST=AUTH

20 When CLASS=TAPEVOL, indicates that the TAPEVOL profile can contain a TVTOC, but currently does not (for a scratch pool volume).

24 When CLASS=TAPEVOL, indicates that the TAPEVOL profile does not contain a TVTOC.

XX If the reason code is greater than or equal to hexadecimal 200 (decimal 512), see "Class descriptor table (CDT) default return codes and reason codes" on page 47.

14 Requested function with STATUS=ACCESS specified has completed successfully. The user's highest access to the specified resource is indicated by one of the following reason codes:

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>The user has no access.</td>
</tr>
<tr>
<td>04</td>
<td>The user has READ authority.</td>
</tr>
<tr>
<td>08</td>
<td>The user has UPDATE authority.</td>
</tr>
<tr>
<td>0C</td>
<td>The user has CONTROL authority.</td>
</tr>
<tr>
<td>10</td>
<td>The user has ALTER authority.</td>
</tr>
</tbody>
</table>

04 Requested function could not be completed. No RACF decision.

RACF RC

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00 No security decision could be made.</td>
</tr>
</tbody>
</table>

Reason code

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00 RACF was not called to process the request because one of the following occurred:</td>
</tr>
<tr>
<td>• RACF is not installed.</td>
</tr>
<tr>
<td>• The combination of class, REQSTOR, and SUBSYS was found in the RACF router table, and ACTION=NONE was specified.</td>
</tr>
<tr>
<td>• The RACROUTE issuer specified DECOUPL=YES and a RELEASE= keyword with a higher release than is supported by this level of z/OS.</td>
</tr>
<tr>
<td>• The specified class is DSNR and the DSNR class is inactive.</td>
</tr>
</tbody>
</table>

04 The specified resource is not protected by RACF.

If PROTECTALL is active, no profile is found, and the user ID whose authority was checked does not have the SPECIAL attribute, RACF returns a return code X'08' instead of a return code X'04' and denies access.

Reason code

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00 One of the following has occurred:</td>
</tr>
<tr>
<td>• There is no RACF profile protecting the resource.</td>
</tr>
<tr>
<td>• RACF is not active.</td>
</tr>
</tbody>
</table>
REQUEST=AUTH

- Specified class is not in the RACF class descriptor table.
- Specified class (other than DSNR) is not active.
- Specified class requires SETROPTS RACLST option to be active and it is not.
- CLASS TEMPDSN was active and the data set is a temporary data set.
- A userid of *BYPASS* has been passed on the authorization check. No profile checking will occur.

04 Indicates STATUS=ERASE was specified and the data set is to be erased when scratched.

582 Reserved.

08 Requested function has failed.

RACF RC

Meaning

08 The user is not authorized by RACF to obtain use of the specified RACF-protected resource.

Reason code

Meaning

00 Indicates a normal completion. A possible cause would be PROTECTALL is active, no profile is found, and the user ID whose authority was checked does not have the SPECIAL attribute.

04 Indicates STATUS=ERASE was specified and the data set is to be erased when scratched.

08 Indicates DSTYPE=T or CLASS=TAPEVOL was specified and the user is not authorized to use the specified volume.

0C For tape data set processing, the user is not authorized to use the data set.

10 Indicates DSTYPE=T or CLASS=TAPEVOL was specified and the user is not authorized to specify TAPELBL=(,BLP).

14 Indicates the user is not authorized to open a non-cataloged data set.

18 Indicates the user is not authorized to issue RACROUTE REQUEST=AUTH when system is in tranquil state (MLQUIET).

1C A user with EXECUTE authority to the data set profile specified ATTR=READ, and RACF failed the access attempt.

20 The user's security label does not dominate that of the resource; it fails security label authorization checking.

24 The user's security label can never dominate that of the resource.

28 The resource must have a security label, but does not have one.

2C Conditional access could not be granted because the environment is not controlled.
If the reason code is greater than or equal to hexadecimal 200 (decimal 512), see "Class descriptor table (CDT) default return codes and reason codes."

0C The OLDVOL specified was not part of the multivolume data set defined by VOLSER, or it was not part of the same tape volume defined by ENTITY.

10 RACROUTE REQUEST=VERIFY was issued by a third party, and RACROUTE REQUEST=AUTH failed.

**Reason code**

**Meaning**

XX This value is the RACF return code from the RACROUTE REQUEST=VERIFY. Refer to "Return codes and reason codes" on page 232 for an explanation of these reason codes. Under SAF return code X'08', see RACF return code XX.

64 Indicates that the CHECK subparameter of the RELEASE keyword was specified on the execute form of the RACROUTE REQUEST=AUTH macro; however, the list form of the macro does not have the same RELEASE parameter. Macro processing terminates.

**Class descriptor table (CDT) default return codes and reason codes**

Normally, if a resource profile is not found, the function returns a return code of 4. However, if a resource profile is not found but a default return-code keyword is specified in the class descriptor table for the class specified on the RACROUTE REQUEST=AUTH, the function returns that specified return code.

When a default return code of other than 4 is specified for a class in the class descriptor table, that specified return code is returned and the reason code is incremented by hexadecimal 200 (decimal 512).

Whenever RACROUTE REQUEST=AUTH returns a default return code, register 1 does not point to a profile, even if CSA or PRIVATE was specified with ENTITY or ENTITYX.

**Example 1**

Perform RACF authorization checking, using the standard form, for a non-VSAM data set residing on the volume pointed to by register 8. Register 7 points to the data set name and the RACF user is requesting the highest level of control over the data set. The “RACF-indicated” bit in the data set’s DSCB is on. Logging and statistics updates are not to be done.

RACROUTE REQUEST=AUTH,ENTITY=((R7)),VOLSER=(R8), X
CLASS='DATASET',WORKA=RACWK, X
ATTR=ALTER,RACFIND=YES,LOG=NOSTAT

: RACWK DS CL512

**Example 2**

Perform RACF authorization checking, using the standard form, for a VSAM data set residing on the volume pointed to by register 8. Register 7 points to the data set name, and the RACF user is requesting the data set for read only. Register 4 points to an area containing additional parameter information.
REQUEST=AUTH

RACROUTE REQUEST=AUTH,ENTITY=((R7)),VOLSER=(R8),  X  
   CLASS='DATASET',WORKA=RACWK,  X  
   DSTYPE=V,INSTLN=(R4)

:  
RACWK   DS   CL512

Example 3
Perform RACF authorization checking, using the standard form, for a tape volume for READ access only. The tape volume is pointed to by register 8 and the volume's access level is in register 5.
RACROUTE REQUEST=AUTH,ENTITY=((R8)),CLASS='TAPEVOL',  X  
   ATTR=READ,WORKA=RACWK,  X  
   ACCLVL=((R5))

:  
RACWK   DS   CL512

Example 4
Perform third-party RACF authorization checking, using the standard form, to provide READ access for a user to a data set residing on the volume pointed to by register 8. Register 7 points to the data set name, and A is an 8-byte declared field padded with blanks.
RACROUTE REQUEST=AUTH,ENTITY=((R7)),WORKA=RACWK,  X  
   VOLSER=(R8),CLASS='DATASET',ATTR=READ,  X  
   USERID=A

:  
RACWK   DS   CL512

Example 5
Perform third-party RACF authorization checking, using the standard form, for a data set for READ access only for a group. Register 8 points to the volume on which the data set resides; register 7 points to the data set name.
RACROUTE REQUEST=AUTH,ENTITY=((R7)),VOLSER=(R8),  X  
   CLASS='DATASET',ATTR=READ,WORKA=RACWK,  X  
   USERID='*NONE*',GROUPID='AGROUPID'

:  
RACWK   DS   CL512

Example 6
Perform third-party RACF authorization checking, using the standard form, for a data set for a user connected to a group. Register 8 points to the volume on which the data set resides; register 7 points to the data set name.
RACROUTE REQUEST=AUTH,ENTITY=((R7)),WORKA=RACWK,  X  
   VOLSER=(R8),CLASS='DATASET',ATTR=READ,  X  
   USERID='SOMEUSER',GROUPID='AGROUPID'

:  
RACWK   DS   CL512

Example 7
Perform third-party RACF authorization checking, using the standard form, for a data set residing on the volume pointed to by register 8, to allow READ access only for a user. Register 7 points to the data set name, and A is an 8-byte declared field padded with blanks.
The list form of the RACROUTE REQUEST=AUTH macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=AUTH macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
</tbody>
</table>

RACROUTE

<table>
<thead>
<tr>
<th>REQUEST=AUTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>,ACCLVL=access level addr</td>
</tr>
<tr>
<td>,ACEE=acee addr</td>
</tr>
<tr>
<td>,APPL=applname</td>
</tr>
<tr>
<td>,APPL=applname addr</td>
</tr>
<tr>
<td>,ATTR=READ</td>
</tr>
<tr>
<td>,ATTR=UPDATE</td>
</tr>
<tr>
<td>,ATTR=CONTROL</td>
</tr>
<tr>
<td>,ATTR=ALTER</td>
</tr>
<tr>
<td>,CLASS=’class name’</td>
</tr>
<tr>
<td>,CLASS=class name addr</td>
</tr>
<tr>
<td>,DSTYPE=N</td>
</tr>
<tr>
<td>,DSTYPE=V</td>
</tr>
<tr>
<td>,DSTYPE=M</td>
</tr>
<tr>
<td>,DSTYPE=T</td>
</tr>
<tr>
<td>,ENTITY=resource name addr</td>
</tr>
<tr>
<td>,ENTITY=(resource name addr)</td>
</tr>
<tr>
<td>,ENTITY=(resource name addr,CSA)</td>
</tr>
<tr>
<td>,ENTITY=(resource name addr,PRIVATE)</td>
</tr>
<tr>
<td>,ENTITY=(resource name addr,NONE)</td>
</tr>
<tr>
<td>,ENTITYX=extended resource name addr</td>
</tr>
</tbody>
</table>
REQUEST=AUTH

,ENTITYX=(extended resource name addr)
,ENTITYX=(extended resource name addr, CSA)
,ENTITYX=(extended resource name addr, PRIVATE)
,ENTITYX=(extended resource name addr, NONE)

,FILESEQ=number

,GENERIC=YES
,GENERIC=ASIS

,GROUPID='groupid'
,GROUPID=groupname addr

,INSTLN=parm list addr

,LOG=ASIS
,LOG=NOFAIL
,LOG=None
,LOG=NOSTAT

,LOGSTR=logstr addr

,OLDVOL=old vol addr

,PROFILE=profile addr

Note: PROFILE, ENTITY, or ENTITYX is required on either the list, execute, or modify form of the macro.

,RACFIND=YES
,RACFIND=NO

,RECVR=recvr addr

,RTOKEN=rtoken addr

,STATUS=NONE
,STATUS=ERASE
,STATUS=EVERDOM
,STATUS=WRITEONLY
,STATUS=ACCESS

,SYSTEM=NO
,SYSTEM=YES

,TAPELBL=STD
,TAPELBL=BLP
,TAPELBL=NL

,USERID='userid'
,USERID=userid addr

Note: Use of the SYSTEM= keyword requires that RELEASE=1.9.2 be specified.
REQUEST=AUTH

,UTOKEN=token addr  token addr: A-type address

,VOLSER=vol addr  vol addr: A-type address

Note: VOLSER is required on either the list or the execute form of the macro for CLASS=DATASET and DSTYPE not equal to M when a discrete profile name is used.

,MF=L

When SYSTEM=YES is specified, only the following keywords are valid:

,CLASS='class name'  class name: 1–8 character name
,CLASS=class name addr  class name addr: A-type address or register (2) – (12)

,ENTITY=resource name addr  resource name addr: A-type address only
,ENTITY=(resource name addr)  resource name addr: A-type address or register (2) – (12)

,ATTR=READ  Default: ATTR=READ
,ATTR=UPDATE
,ATTR=CONTROL
,ATTR=ALTER

,APPL='applname'  applname: 1–8 character name
,APPL=applname addr  applname addr: A-type address or register (2) – (12)

,INSTLN=parm list addr  parm list addr: A-type address or register (2) – (12)

,LOG=ASIS  Default: LOG=ASIS
,LOG=NFAIL
,LOG=NONE
,LOG=NOSTAT

The parameters are explained under the standard form of the RACROUTE REQUEST=AUTH macro with the following exception:

, ,MF=L

specifies the list form of the RACROUTE REQUEST=AUTH macro instruction.

RACROUTE REQUEST=AUTH (execute form)

The execute form of the RACROUTE REQUEST=AUTH macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=AUTH macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
</tbody>
</table>

RACROUTE

| b    | One or more blanks must follow RACROUTE. |

REQUEST=AUTH
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.ACCLVL=access level addr</td>
<td>access level addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.ACEE=acee addr</td>
<td>acee addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.APPL=applname addr</td>
<td>applname addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.ATTR=READ</td>
<td></td>
</tr>
<tr>
<td>.ATTR=UPDATE</td>
<td></td>
</tr>
<tr>
<td>.ATTR=CONTROL</td>
<td></td>
</tr>
<tr>
<td>.ATTR=ALTER</td>
<td></td>
</tr>
<tr>
<td>.ATTR=reg</td>
<td>reg: Register (2) – (12)</td>
</tr>
<tr>
<td>.CLASS=class name addr</td>
<td>class name addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.DSTYPE=N</td>
<td></td>
</tr>
<tr>
<td>.DSTYPE=V</td>
<td></td>
</tr>
<tr>
<td>.DSTYPE=M</td>
<td></td>
</tr>
<tr>
<td>.DSTYPE=T</td>
<td></td>
</tr>
<tr>
<td>.ENTITY=resource name addr</td>
<td>resource name addr: Rx-type address only</td>
</tr>
<tr>
<td>.ENTITY=(resource name addr)</td>
<td>resource name addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.ENTITY=(resource name addr,CSA)</td>
<td></td>
</tr>
<tr>
<td>.ENTITY=(resource name addr,PRIVATE)</td>
<td></td>
</tr>
<tr>
<td>.ENTITY=(resource name addr,NONE)</td>
<td></td>
</tr>
<tr>
<td>.ENTITYX=extended resource name addr</td>
<td>extended resource name addr: Rx-type address only</td>
</tr>
<tr>
<td>.ENTITYX=(extended resource name addr)</td>
<td>extended resource name addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.ENTITYX=(extended resource name addr,CSA)</td>
<td></td>
</tr>
<tr>
<td>.ENTITYX=(extended resource name addr,PRIVATE)</td>
<td></td>
</tr>
<tr>
<td>.ENTITYX=(extended resource name addr,NONE)</td>
<td></td>
</tr>
<tr>
<td>.FILESEQ=number</td>
<td>number: 1-65535</td>
</tr>
<tr>
<td>.FILESEQ=reg</td>
<td>reg: Register (2) – (12)</td>
</tr>
<tr>
<td>.GENERIC=YES</td>
<td></td>
</tr>
<tr>
<td>.GENERIC=ASIS</td>
<td></td>
</tr>
<tr>
<td>.GROUPID=groupname addr</td>
<td>groupname addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.INSTLN=parm list addr</td>
<td>parm list addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.LOG=ASIS</td>
<td></td>
</tr>
<tr>
<td>.LOG=NOFAIL</td>
<td></td>
</tr>
<tr>
<td>.LOG=None</td>
<td></td>
</tr>
<tr>
<td>.LOG=NOSTAT</td>
<td></td>
</tr>
</tbody>
</table>
REQUEST=AUTH

,LOGSTR=LOGSTR addr
,OLDVOL=OLDVOL addr
,PROFILE=PROFILE addr

Note: PROFILE, ENTITY, or ENTITYX required on either the list, execute, or modify form of the macro.

,RACFIND=YES
,RACFIND=NO

,RECVR=RECVR addr

,RELEASE=number
,RELEASE=(CHECK)

,RTOKEN=RTOKEN addr

,STATUS=NONE
,STATUS=ERASE
,STATUS=EVERDOM
,STATUS=WRITEONLY

,SYSTEM=NO
,SYSTEM=YES

,TAPELBL=STD
,TAPELBL=BLP
,TAPELBL=NL

,USERID=USERID addr

,UTOKEN=UTOKEN addr

,VOLSER=VOLSER addr

Note: VOLSER is required on either the list or the execute form of the macro for CLASS=DATASET and
DSTYPE not equal to M when a discrete profile name is used.

,MF=(E,ctrl addr)  ctrl addr: Rx-type address, or register (1) or (2) – (12)

When SYSTEM=YES is specified, only the following keywords are valid:

,CLASS=CLASS addr
,ENTITY=ENTITY addr

,ATTR=READ
,ATTR=UPDATE
,ATTR=CONTROL
,ATTR=ALTER
,ATTR=reg

,APPL=APPL addr

Chapter 3. System macros  53
REQUEST=AUTH

 parm list addr: A-type address or register (2) – (12)

LOG=ASIS
LOG=NONE
LOG=NOFAIL
LOG=NOSTAT

The parameters are explained under the standard form of the RACROUTE REQUEST=AUTH macro with the following exceptions:

\textbf{RELEASE}\texttt{=number}

\textbf{RELEASE}\texttt{=(,CHECK)}

\textbf{RELEASE}\texttt{=(number,CHECK)}

specifies the RACF release level of the parameter list to be generated by this macro.

To use the parameters associated with a release, you must specify the release number of that release or a later release number. If you specify an earlier release level, the parameter is not accepted by macro processing, and an error message is issued at assembly time.

When you specify the RELEASE keyword, checking is done at assembly time. Compatibility between the list and execute forms of the RACROUTE REQUEST=AUTH macro is verified at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.

The default is RELEASE=1.6.

\textbf{MF}\texttt{=(E,ctrl addr)}

specifies the execute form of the RACROUTE REQUEST=AUTH macro instruction.

\textbf{RACROUTE REQUEST=AUTH (modify form)}

The modify form of the RACROUTE REQUEST=AUTH macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=AUTH macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

\texttt{name}

\textit{name}: Symbol. Begin \texttt{name} in column 1.

\texttt{b}

One or more blanks must precede RACROUTE.

RACROUTE

\texttt{b}

One or more blanks must follow RACROUTE.

REQUEST=AUTH

\texttt{ACCLVL}\texttt{=access level addr}

\textit{access level addr}: Rx-type address or register (2) – (12)
REQUEST=AUTH

.ACCE=acee addr
.ACCE=Rx-type address or register (2) – (12)

.APPL=applname addr
.APPL=Rx-type address or register (2) – (12)

.ATTR=READ
.ATTR=UPDATE
.ATTR=CONTROL
.ATTR=ALTER
.ATTR=reg
.reg: Register (2) – (12)

.CLASS=class name addr
.class name addr: Rx-type address or register (2) – (12)

.DSTYPE=N
.DSTYPE=V
.DSTYPE=M
.DSTYPE=T

.ENTITY=resource
.resource name addr: Rx-type address only

.ENTITY=(resource
.resource name addr)

.ENTITY=(resource
.resource name addr,CXA)

.ENTITY=(resource
.resource name addr,PRIVATE)

.ENTITY=(resource
.resource name addr,NONE)

.ENTITYX=extended
.extended resource name addr: Rx-type address only

.ENTITYX=(extended
.extended resource name addr)

.ENTITYX=(extended
.extended resource name addr,CXA)

.ENTITYX=(extended
.extended resource name addr,PRIVATE)

.ENTITYX=(extended
.extended resource name addr,NONE)

.FILESEQ=number
.number: 1-65535

.FILESEQ=reg
.reg: Register (2) – (12)

.GENERIC=YES
.GENERIC=ASIS

.GROUPID=groupname
.groupname addr: Rx-type address or register (2) – (12)

.INSTLN=parm list addr
.parm list addr: Rx-type address or register (2) – (12)

.LOG=ASIS
.LOG=NOFAIL
.LOG=NONE
.LOG=NOSTAT

.LOGSTR=logstr addr
.logstr addr: Rx-type address or register (2) – (12)
REQUEST=AUTH

,OLDVOL=old vol addr   old vol addr: Rx-type address or register (2) – (12)

,PROFILE=profile addr   profile addr: Rx-type address or register (2) – (12)

Note: PROFILE, ENTITY, or ENTITYX is required on either the list, execute, or modify form of the macro.

,RACFIND=YES
,RACFIND=NO

,RECVR=recvr addr      recvr addr: Rx-type address or register (2) – (12)

,RTOKEN=rtoken addr    rtoken addr: Rx-type address or register (2) – (12)

,STATUS=None
,STATUS=erase
,STATUS=everdom
,STATUS=writeonly
,STATUS=access

,SYSTEM=NO            Default: SYSTEM=NO
,SYSTEM=YES

Note: Use of the SYSTEM= keyword requires that RELEASE=1.9.2 be specified.

,TAPELBL=STD
,TAPELBL=BLP
,TAPELBL=NL

,USERID=userid addr    userid addr: Rx-type address or register (2) – (12)

,UTOKEN=token addr     token addr: Rx-type address or register (2) – (12)

,VOLSER=vol addr       vol addr: Rx-type address or register (2) – (12)

Note: VOLSER is required on either the list or the execute form of the macro for CLASS=DATASET and DSTYPE not equal to M when a discrete profile name is used.

,MF=(M,ctrl addr)      ctrl addr: Rx-type address, or register (1) or (2) – (12)

When SYSTEM=YES is specified, only the following keywords are valid:

,CLASS=class name addr          class name addr: A-type address or register (2) – (12)

,ENTITY=resource name addr     resource name addr: A-type address only
,ENTITY=(resource name addr)   resource name addr: A-type address or register (2) – (12)

,ATTR=READ
,ATTR=UPDATE
,ATTR=CONTROL
,ATTR=ALTER
,ATTR=REG

,APPL=applname addr        applname addr: A-type address or register (2) – (12)

,INSTLN=parm list addr      parm list addr: A-type address or register (2) – (12)

,LOG=ASIS
,LOG=NOFAIL
,LOG=NONE
REQUEST=AUTH

The parameters are explained under the standard form of the RACROUTE REQUEST=AUTH macro with the following exceptions:

\( MF=(M, ctrl\ addr) \)

specifies the modify form of the RACROUTE REQUEST=AUTH macro instruction.

---

**RACROUTE REQUEST=DEFINE: Define, modify, rename, or delete a resource for RACF**

The RACROUTE REQUEST=DEFINE macro defines, modifies, renames, or deletes resource profiles for RACF. You can also use it for special cases of authorization checking. RACF uses the resulting profiles to perform authorization checking when a user requests access to a RACF-protected resource.

In general, you should use the RACF command processors to create RACF resource profiles, because only the command processors do complete validation of profile name syntax. If you use RACROUTE REQUEST=DEFINE instead, you should create profiles which are supported by the command processors. For instance, RACROUTE REQUEST=DEFINE allows you to create a fully qualified generic profile in a general resource class and a data set profile containing characters that are not valid, but those profiles are not supported by the RACF command processors.

The RACROUTE REQUEST=DEFINE preprocessing and postprocessing exit routines can change or add the RACROUTE REQUEST=DEFINE parameters OWNER, LEVEL, UACC, or AUDIT.

The RACROUTE REQUEST=DEFINE caller must be authorized (APF-authorized, in system key 0–7, or in supervisor state).

The caller cannot hold any locks when issuing RACROUTE REQUEST=DEFINE.

When activated, automatic direction of application updates propagates RACROUTE REQUEST=DEFINE updates to selected remote nodes.

Not all RACROUTE REQUEST=DEFINE requests update the RACF database. The ENVIR=VERIFY keyword specifies that no profile is to be created, but that the user’s authority is to be checked. Automatic direction of application updates does not propagate a RACROUTE REQUEST=DEFINE if ENVIR=VERIFY is specified. Similarly, many RACROUTE REQUEST=DEFINE requests are issued with RACFIND=NO to check if a user is authorized to create a data set or catalog a data set based on a generic profile. RACROUTE REQUEST=DEFINE RACFIND=NO requests for DASD data sets are not propagated. RACROUTE REQUEST=DEFINE requests for tape data sets are propagated even when RACFIND=NO is specified because even though no data set profile is updated, an update might be made to the TVTOC in a TAPEVOL profile.

Only RACROUTE REQUEST=DEFINE requests with return code 0 are propagated.
REQUEST=DEFINE

RACROUTE REQUEST=DEFINE (standard form)

The standard form of the RACROUTE REQUEST=DEFINE macro is written as follows. For a description of additional keywords that you can code and additional parameters that are required on the RACROUTE request, but that are not specific to this request type, see "RACROUTE (standard form)" on page 13.

Note:

RACROUTE REQUEST=DEFINE requires an ACEE. For most applications, the system will have created an ACEE to represent the active user. However, for special cases where no ACEE exists, the invoker must create one before invoking RACROUTE REQUEST=DEFINE.

The most common way to create an ACEE is to issue a RACROUTE REQUEST=VERIFY, specifying ENVIR=CREATE. After all RACROUTE invocations are complete, the invoker should issue RACROUTE REQUEST=VERIFY with ENVIR=DELETE specified, to delete the ACEE previously created.

Application programs must be structured so that a task requesting RACF services does not do so while other I/O initiated by the task is outstanding. If such I/O is required, the task should either wait for the other I/O to complete before requesting RACF services, or the other I/O should be initiated under a separate task. This is necessary to assure proper processing in recovery situations.

name

name: Symbol. Begin name in column 1.

b

One or more blanks must precede RACROUTE.

RACROUTE

b

One or more blanks must follow RACROUTE.

REQUEST=DEFINE

,ENTITY=profile name addr

profile name addr: A-type address or register (2) – (12)

,ENTITYX=extended profile name addr

extended profile name addr: A-type address or register (2) – (12)

,VOLSER=vol addr

vol addr: A-type address or register (2) – (12)

Note: VOLSER is required for CLASS=DATASET and DSTYPE not equal to M when a discrete profile name is used.

,ACCLVL=access value addr

access value addr: A-type address or register (2) – (12)

,ACCLVL=(access value addr,parm list addr)

parm list addr: A-type address or register (2) – (12)

,ACEE=acee addr

acee addr: A-type address or register (2) – (12)

,AUDIT=NONE

Note: AUDIT is valid if TYPE=DEFINE is specified.

,AUDIT=audit value

audit value: ALL, SUCCESS, or FAILURES

,AUDIT=(audit value,access level),audit value (access level)
REQUEST=DEFINE

, AUDIT=reg
  Default: AUDIT=READ
  reg: register (2) – (12)

, CHKAUTH=YES, CHKAUTH=NO
  Default: CHKAUTH=NO

, CLASS=class name
  class name: 1–8 character name.

, CLASS=class name addr
  class name addr: A-type address or register (2) – (12)
  Default: CLASS=DATASET

, DATA=data addr
  data addr: A-type address or register (2) – (12)

, DSTYPE=N, DSTYPE=V, DSTYPE=M, DSTYPE=T
  Default: DSTYPE=N

, ENVIR=VERIFY
  Specifies that only verification is to be done.
  Default: Normal RACROUTE REQUEST=DEFINE processing

, ERASE=YES, ERASE=NO
  Default: ERASE=NO

, EXPDT=expir-date addr
  expir-date addr: A-type address or register (2) – (12)

, EXPDTX=extended expir-date addr
  extended expir-date addr: A-type address or register (2) – (12)

, RETPD=retn-period addr
  retn-period addr: A-type address or register (2) – (12)
  Default: See description of parameter.

, FILESEQ=number, FILESEQ=reg
  number: 1–65535
  reg: Register (2) – (12)

, GENERIC=YES, GENERIC=ASIS
  Default: GENERIC=ASIS

, INSTLN=parm list addr
  parm list addr: A-type address or register (2) – (12)

, LEVEL=number, LEVEL=reg
  Default: LEVEL=zero
  reg: Register (2) – (12)

, MCLASS=class name
  class name: 1–8 character name

, MCLASS=class name addr
  class name addr: A-type address or register (2) – (12)
  Default: MCLASS=DATASET

, MENTITY=entity addr, MENTITY=extended entity addr
  entity addr: A-type address or register (2) – (12)
  extended entity addr: A-type address or register (2) – (12)

, MGENER=ASIS, MGENER=YES
  Default: MGENER=ASIS

, MGMTCLA=management type addr
  management type addr: A-type address or register (2) – (12)

, MVOLSER=volser addr
  volser addr: A-type address or register (2) – (12)

, NOTIFY=notify-id addr
  notify-id addr: A-type address or register (2) – (12)
REQUEST=DEFINE

,OWNER=owner id addr
owner id addr: A-type address or register (2) – (12)

,RACFIND=YES
,RACFIND=NO

,RESOWN=resource owner addr
resource owner addr: A-type address or register (2) – (12)

,SECLABL=addr
addr: A-type address or register (2) – (12)

,SECLVL=addr
addr: A-type address or register (2) – (12)

,STORCLA=storage class addr
storage class addr: A-type address or register (2) – (12)

,TAPELBL=STD
Default: TAPELBL=STD

,TAPELBL=BLP

,TAPELBL=NL

,TYPE=DEFINE
,TYPE=DEFINE,
NEWWNAME=new resource name addr
new resource name addr: A-type address or register (2) – (12)

,TYPE=DEFINE,
NEWNAMX=extended new resource name addr
extended new resource name addr: A-type address or register (2) – (12)

,TYPE=DEFINE,
NEWNAMX=extended new resource name addr

,OLDVOL=old vol addr
old vol addr: A-type address or register (2) – (12)

,ADDVOL=old vol addr

,CHGVOl=old vol addr

,DELETE

,ALTER

,UPDATE

,READ

,EXECUTE

,NONE

,REG=reg
reg: Register (2) – (12)

,UNIT=unit addr
unit addr: A-type address or register (2) – (12)

,WARNING=YES
Default: WARNING=NO

,WARNING=NO

Note: WARNING is valid if TYPE=DEFINE is specified.

,MF=S

The parameters are explained as follows:

,ENTITY=profile name addr
,ENTITYX=extended profile name addr
specifies the address:
REQUEST=DEFINE

- ENTITY=profile name addr specifies the address of the name of the discrete or generic profile that is to be defined to, modified, or deleted from RACF. The profile name is a 44-byte DASD data set name for CLASS=DATASET, or a 6-byte volume serial name for CLASS=DASDVOL or CLASS=TAPEVOL. The lengths of all other profile names are determined by the class descriptor table. The name must be left-justified in the field and padded with blanks.

- ENTITYX=extended profile name addr specifies the address of a structure that consists of two 2-byte length fields, followed by the entity name.
  - The first 2-byte field specifies a buffer length which can be from 0 to 255 bytes. This length field refers to the length of the buffer that contains the entity name; it does not include the length of either length field.
  - The second 2-byte field specifies the actual length of the entity name. This length field includes the length of the actual name without any trailing blanks; it does not include the length of either length field.

These two length fields can be used in several different ways:
- If you know the length of the entity name, you can specify 0 in the first field and the length of the entity name in the second field. When you specify the second field, note that each byte counts. This means the entity name that you specify must match exactly the entity name on the RACF database.

- If you choose to place the entity name in a buffer area, specify the length of the buffer in the first field. For the second field, do one of the following:
  - If you know the length of the entity name, specify the length in the second field. The length of the first field can be from 0 to 255, but must be equal to or greater than the length of the second field.
  - If you do not know the length of the entity name, specify 0 in the second field, and have RACF determine the number of characters in the entity name.

To use this keyword, you must specify RELEASE=1.9 or a later release number.

In general, you should use the RACF command processors to create RACF resource profiles, since only the command processors do complete validation of profile name syntax. If you use RACROUTE REQUEST=DEFINE instead, you should create profiles which are supported by the command processors. For instance, RACROUTE REQUEST=DEFINE allows you to create a fully qualified generic profile in a general resource class and a data set profile containing characters that are not valid, but those profiles are not supported by the RACF command processors.

Guideline: Use ENTITYX rather than ENTITY. With ENTITY, the entity name you pass to RACF must be in a buffer, the size of which is determined by the length in the class descriptor table. If the maximum length of a class descriptor entity increases in the future, you must modify your program to use a larger buffer. By using ENTITYX, you avoid this possible problem because you have removed the class descriptor table dependency from your program.

\,VOLSER=vol addr

specifies the address of the volume serial number:
- For TYPE=ADDVOL, of the new volume to be added to the definition of the data set
- For TYPE=ADDVOL and CLASS=TAPEVOL, of the new volume being added to the tape volume set identified by ENTITY or ENTITYX
REQUEST=DEFINE

- For TYPE=DEFINE and CLASS=DATASET, of the catalog (for a VSAM data set), or of the volume on which the data set resides (for a non-VSAM data set).

The volume serial number is optional if DSTYPE=M is specified; it is ignored if the profile name is generic.

The field pointed to by the specified address contains the volume serial number, padded to the right with blanks if necessary to make six characters.

,ACCLVL=access value addr
,ACCLVL=(access value addr, parm list addr)
specifies the tape-label access-level information for the MVS tape-label functions. The address must point to a field containing a one-byte length field (with a value from 0–8) followed by an 8-character string that is passed to the RACROUTE REQUEST=DEFINE installation exit routines. The parameter list address points to a parameter list containing additional information to be passed to the RACROUTE REQUEST=DEFINE installation exit routines.

RACF does not check or modify this information.

,ACEE=acee addr
specifies the address of the ACEE to be used during RACROUTE REQUEST=DEFINE processing.

If no ACEE is specified, RACF uses the TASK ACEE pointer (TCBSENV) in the extended TCB. If the TASK ACEE pointer is zero, RACF uses the main ACEE. The main ACEE’s address is in the ASXBSENV field in the address-space extension block.

,AUDIT=None
,AUDIT=audit value
,AUDIT=(audit value(access level), audit value(access level), ...)
,AUDIT=reg
specifies the types of accesses and the access levels that are to be logged to the SMF data set.

For audit value, specify one of the following: ALL, SUCCESS, or FAILURES. You can optionally specify an access level(access authority) following each audit value.

Access Levels:
- EXECUTE is not audited. If you specify FAILURES (READ), RACF logs the READ attempt as a failure, but allows EXECUTE access to the data set.
- READ logs access attempts at any level. READ is the default access-level value.
- UPDATE logs access attempts at the UPDATE, CONTROL, and ALTER levels.
- CONTROL logs access attempts at the CONTROL and ALTER levels.
- ALTER logs access attempts at the ALTER level only.

Note: For more information about specific audit values and access levels, see z/OS Security Server RACF Command Language Reference.

RACF resolves combinations of conflicting specifications by using the most encompassing specification. Thus, in the case of the following:
ALL(UPDATE), FAILURES(READ)
RACF assumes SUCCESS(UPDATE), FAILURES(READ).
REQUEST=DEFINE

For compatibility with previous releases, register notation can also be specified as AUDIT=reg if the register is not given as a symbolic name; for example, ALL, SUCCESS, or FAILURES.

Logging is controlled separately for SUCCESS and FAILURES, and can also be suppressed or requested using the RACROUTE REQUEST=AUTH postprocessing installation exit routine.

If a register is specified, its low-order byte must contain one of the following valid audit values:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ALL</td>
</tr>
<tr>
<td>1</td>
<td>SUCCESS</td>
</tr>
<tr>
<td>2</td>
<td>FAILURES</td>
</tr>
<tr>
<td>3</td>
<td>NONE</td>
</tr>
<tr>
<td>4-5</td>
<td>Qualifier for SUCCESS</td>
</tr>
<tr>
<td>6-7</td>
<td>Qualifier for FAILURES</td>
</tr>
</tbody>
</table>

The qualifier codes are as follows:

<table>
<thead>
<tr>
<th>Code</th>
<th>Qualifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>READ</td>
</tr>
<tr>
<td>01</td>
<td>UPDATE</td>
</tr>
<tr>
<td>10</td>
<td>CONTROL</td>
</tr>
<tr>
<td>11</td>
<td>ALTER</td>
</tr>
</tbody>
</table>

Only one of bits 0 through 3 can be on. If ALL is specified, the two qualifier fields can be used to request different logging levels for successful and unsuccessful events.

**Note:** RACF does not check the validity of the audit type if it has been added or modified by the RACROUTE REQUEST=DEFINE preprocessing or postprocessing exit routine.

AUDIT is valid if TYPE=DEFINE is specified.

,CHKAUTH=YES
,CHKAUTH=NO

specifies whether or not an internal RACROUTE REQUEST=AUTH with ATTR=ALTER is to be done to verify that the user is authorized to perform the operation.

CHKAUTH=YES is valid when TYPE=DEFINE,NEWWNAME or TYPE=DEFINE,NEWNAMX, or TYPE=DELETE is specified.

For DSTYPE=T, CHKAUTH=YES specifies that an internal RACROUTE REQUEST=AUTH with ATTR=UPDATE be done to verify that the user is authorized to define a data set (TYPE=DEFINE), delete a data set (TYPE=DELETE), or add a volume (TYPE=ADDVOL).

The default is CHKAUTH=NO.

,CLASS='class name'
,CLASS=class name addr

specifies that a profile is to be defined, modified, or deleted in the specified class. If an address is specified, the address must point to a one-byte length
REQUEST=DEFINE

field followed by the class name (such as DATASET or TAPEVOL). The class
name should be no longer than 8 characters.

Note: Do not use RACGLIST as the class name.

\_DATA=data addr

specifies the address of a field that contains up to 255 characters of
installation-defined data to be placed in the profile. The data address must
point to a field containing a one-byte length field (whose value can range from
0 to 255) followed by the actual installation-defined data.

DATA is valid if TYPE=DEFINE is specified.

\_DSTYPE=N
\_DSTYPE=V
\_DSTYPE=M
\_DSTYPE=T

specifies the type of data set associated with the request:

N for non-VSAM
V for VSAM
M for model profile
T for tape

DSTYPE=T should not be specified unless the SETROPTS TAPEDSN option is
active (RCVTXTDSN bit is on); otherwise, the processing is the same as for
RACROUTE REQUEST=DEFINE, CLASS='TAPEVOL'.

Specify DSTYPE only when the value of CLASS is DATASET.

\_ENVIR=VERIFY

specifies that no profile is to be created, but that the user's authority to define
or rename the resource or profile is to be checked, along with any other
authorization processing that is necessary.

If you specify ENVIR, you must specify RELEASE=1.8.1 or a later release
number.

Note:
1. If you do not specify ENVIR=VERIFY, normal RACROUTE
   REQUEST=DEFINE processing occurs.
2. Automatic direction of application updates does not propagate a
   RACROUTE REQUEST=DEFINE request if ENVIR=VERIFY is specified.

\_ERASE=YES
\_ERASE=NO

specifies whether the DASD data set, or the released space, is to be erased
when it is deleted or part of its space is to be released for reuse.

If ERASE=YES is specified, the data set is erased when it is deleted or released
for reuse.

If ERASE=NO is specified, the data set is not erased, deleted, or released.

The default is ERASE=NO.

Specify ERASE only for CLASS=DATASET.

Note: This parameter can be overridden by the RACF SETROPTS ERASE
command.
 REQUEST=DEFINE

,EXPDT=expir-date addr
,EXPDTX=extended expir-date addr
,RETPD=retn-period addr

specifies the address containing information about the expiration date or RACF security retention period of the data set.

- EXPDT=expir-date addr specifies the address of a 3-byte field containing the expiration date of the data set. The date is given in packed decimal form as YYDDDF, where YY is the year and DDD is the day number. The year must be in the range 00 through 99. The day number must be in the range 0 through 366. The year is treated as 19YY. To specify a year in the range 2000–2155, you must use EXPDTX instead of EXPDT. F allows the date to remain a positive integer when converted from packed decimal to hexadecimal. All fields are right-justified.

**Note:** Processing of this keyword depends on the specification of the 2DGT_EXPDT statement in the z/OS SYS1.PARMLIB member ALLOCxx:

- POLICY(ALLOW) results in RACF processing the EXPDT keyword as documented. Using the EXPDT keyword after January 1, 2000 with other than the values 99365 or 99366 results in a data set definition that expires after the default retention period as set by SETROPTS RETPD.
- POLICY(WARN) results in RACF processing EXPDT as above. In addition, RACF issues message ICH902I, which states that you cannot specify dates beyond 1999 with this keyword.
- POLICY(FAIL) results in RACF failing the request and issuing message, ICH903I, which indicates that a 3-byte expiration date cannot be specified. Instead, you must use the keyword EXPDTX, which specifies a 4-byte expiration date.

The values 99365, 99366, 99000, and 98000 are exempted from both the WARN and FAIL policy settings. RACF treats 99365 and 99366 as “never expire” dates. The 99000 and 98000 dates are allowed, and might be special to some OEM tape management products. Because these dates are earlier than the current date, RACF does not use them to determine the retention period for the data set. It uses the default retention period set by the SETROPTS RETPD command.

See [z/OS MVS Initialization and Tuning Reference](https://www.ibm.com/docs/en/zos/) and the documentation associated with APAR OW39170 for more information on ALLOCxx.

- EXPDTX=extended expir-date addr specifies the address of a 4-byte field that contains the expiration date of the data set. The date is given in packed decimal form as CCYYDDDF, where CC is 00 for years in the range 1900–1999, 01 for years in range 2000–2099, and 02 for years in the range 2100–2155. YY is the year, and DDD is the day number. The year must be in the range 00 through 99. The day must be in the range 1 through 366. The combined CCYY value cannot specify a year greater than 2155, so 0255 is the maximum value that can be specified for the combined CCYY field. All fields are right-justified. F allows the date to remain positive when converted from packed decimal to hexadecimal.

**Note:** Specifying 99365 or 99366 for YYDDDF on EXPDT, or 0099365 or 0099366 for CCYYDDDF on EXPDTX indicates an expiration date of NEVER EXPIRES. This means that an actual expiration date of 12/31/1999 cannot be specified using either of these keywords. Use the RETPD keyword with the appropriate value if that expiration date is desired.

- RETPD=retn-period addr specifies the address of a 2-byte binary field containing the number of days after which RACF protection for the data set...
REQUEST=DEFINE

expires. The value specified must be in the range 1 through 65533. To indicate that there is no expiration date, specify 65534.

If you do not specify any of these parameters, or if the date given on EXPDT or EXPDTX is prior to the current date, a default RACF security retention period is obtained from the RETPD keyword specified on an earlier RACF SETROPTS command.

These parameters are valid if CLASS=DATASET and DSTYPE=T.

**FILESEQ=number**

specifies the file sequence number of a tape data set on a tape volume or within a tape-volume set. The number must be in the range 1–65535. If a register is specified, it must contain the file sequence number in the low-order halfword. If CLASS=DATASET and DSTYPE=T are not specified, FILESEQ is ignored.

**FILESEQ=reg**

**GENERIC=YES**

**GENERIC=ASIS**

specifies whether the resource name is treated as a generic profile name. If GENERIC is specified with CLASS=DEFINE, NEWNAME, or NEWNAMX, GENERIC applies to both the old and new names. GENERIC is ignored if the GENCMD option on the RACF SETROPTS command is not specified for the class. See z/OS Security Server RACF Command Language Reference.

This keyword is designed primarily for use by RACF commands.

**YES**

The resource name is considered a generic profile name, even if it does not contain a generic character: an asterisk (*), a percent sign (%), or, for general-resource classes, an ampersand sign (&). If you specify GENERIC=YES, the resource name in the macro will match only a generic resource name that is exactly the same in the RACF data base. It will not match a discrete name.

**ASIS**

The resource name is considered a generic if it contains a generic character: an asterisk (*), a percent sign (%), or, for general resource classes, an ampersand sign (&).

**INSTLN=parm list addr**

specifies the address of an area that is to contain parameter information meaningful to the RACROUTE REQUEST=DEFINE installation exit routines. This information is passed to the installation exit routines when they are given control from the RACROUTE REQUEST=DEFINE routine.

The INSTLN parameter can be used by an application program acting as a resource manager that needs to pass information to the RACROUTE REQUEST=DEFINE routine.

**LEVEL=number**

**LEVEL=reg**

specifies a level value for the profile. The level number must be a valid decimal number in the range 0 to 99. If a register is specified, its low-order byte must contain the binary representation of the number.

LEVEL is valid if TYPE=DEFINE is specified.

**MCLASS='class name'**

**MCLASS=class name addr**

specifies the class to which the profile defined by MENTITY= or MENTX= belongs. If an address is specified, the address must point to a 1-byte length
REQUEST=DEFINE

field followed by the class name. The class name should be no longer than 8 characters. The default is MCLASS=DATASET.

`,MVOLSER=volser addr`
specifies the address of the volume serial number of the volume associated with the profile in the MENTITY operand. The field pointed to by the specified address contains the volume serial number, padded to the right with blanks if necessary to make six characters.

If you specify MENTITY or MENTX and CLASS=DATASET, you must specify MVOLSER with the name of the VOLSER or with blanks.

If you specify with blanks, the discrete MENTITY or MENTX data set profile name must be unique, meaning it has no duplicates on the database. In this case, RACF determines the correct MVOLSER.

`,MENTITY=entity addr`
`,MENTX=extended entity address`
specifies the address of the name of the discrete or generic profile that is to be used:

- MENTITY=entity addr specifies the address of the name of the discrete or generic profile that is to be used as a model in defining the ENTITY or ENTITYX profile. The profile can belong to any class, as specified by the MCLASS parameter, and can be either a discrete or a generic profile.

  MENTITY can be specified with TYPE=DEFINE but not with TYPE=DEFINE,NEWNAME=new resource name addr.

  For data sets, the name is contained in a 44-byte field pointed to by the specified address. For general resource classes, the length of the field is determined by the class descriptor table. The name is left-justified in the field and padded with blanks.

- MENTX=extended entity address specifies the address of the name of the discrete or generic profile that is to be used as a model from which to define the ENTITY or ENTITYX profile. The structure consists of two 2-byte length fields, followed by the entity name.

  - The first 2-byte field specifies a buffer length that can be from 0 to 255 bytes. This length field refers to the length of the buffer that contains the entity name from which you are modeling; it does not include the length of either length field.

  - The second 2-byte field specifies the actual length of the entity name from which you are modeling. This length field includes the length of the actual name without any trailing blanks; it does not include the length of either length field.

These two length fields can be used in several different ways:

- If you know the length of the entity name you are using as a model, you can specify 0 in the first field and the length of the entity name in the second field. When you specify the second field, note that each byte counts. This means the entity name that you specify is used as a model, using the specified length.

- If you choose to place the entity name in a buffer area, specify the length of the buffer in the first field. For the second field, do one of the following:

  - If you know the length of the entity name you are using as a model, specify the length in the second field. The length of the first field can be from 0 to 255, but must be equal to or greater than the length of the second field.
REQUEST=DEFINE

- If you do not know the length of the entity name you are using as a model, specify 0 in the second field, and have RACF determine the number of characters in the entity name.

To use this keyword, you must specify RELEASE=1.9 or a later release number.

**Guideline:** Use MENTX rather than MENTITY. With MENTITY, the entity name you pass to RACF must be in a buffer, the size of which is determined by the length in the class descriptor table. If the maximum length of a class descriptor entity increases in the future, you must modify your program to use a larger buffer. By using MENTX, you avoid this possible problem, because you remove the class descriptor table dependency from your program.

The profile can belong to any class, as specified by the MCLASS parameter, and can be either a discrete or generic profile. MENTX can be specified with TYPE=DEFINE, but not with TYPE=DEFINE,NEWNAME= or TYPE=DEFINE,NEWNAMX=.

\$MGENER=ASIS
$MGENER=YES

specifies whether the profile name defined by MENTITY or MENTX is to be treated as a generic name.

**ASIS** The profile name is considered a generic if it contains a generic character: an asterisk (*), a percent sign (%), or, for general resource classes, an ampersand sign (&).

**YES** The resource name is considered a generic profile name, even if it does not contain a generic character: an asterisk (*), a percent sign (%), or, for general-resource classes, an ampersand sign (&). If you specify GENERIC=YES, the resource name in the macro will match only a generic resource name in the RACF data base. It will not match a discrete name.

MGENER is ignored if the GENCMD option on the RACF SETROPTS command is not specified for the class. See z/OS Security Server RACF Command Language Reference.

\$MGMTCLA=management type addr

specifies the address of a management class to which the resource owner must have authority. The address must point to an 8-byte field that contains a management-class name preceded by a halfword length. If you specify MGMTCLA, you must specify TYPE=DEFINE, RESOWN, and RELEASE=1.8.1 or a later release number.

When MGMTCLA is specified, RACROUTE REQUEST=DEFINE processing invokes REQUEST=AUTH processing to verify that the RESOWNER is authorized to the management class.

\$NOTIFY=notify-id addr

specifies the address of an 8-byte area containing the user ID of the RACF-defined user who is to be notified when an unauthorized attempt to access the resource protected by this profile is detected.

\$OWNER=owner id addr

specifies the address of a field containing the profile owner's ID. The owner's ID must be a valid (RACF-defined) user ID or group name. The address must point to an 8-byte field containing the owner's name, left-justified and padded with blanks.

OWNER is valid if TYPE=DEFINE is specified.
REQUEST=DEFINE

,RACFIND=YES
,RACFIND=NO

specifies whether or not a discrete profile is involved in RACROUTE REQUEST=DEFINE processing.

When TYPE=DEFINE is specified, RACFIND=YES means that a discrete profile is to be created. When the request type TYPE=DELETE, DEFINE with NEWNAME or NEWNAMX, CHGVOL, or ADDVOL is specified, RACFIND=YES means that a discrete profile already exists. The bit on the VTOC is ignored.

When TYPE=DEFINE is specified, RACFIND=NO means that no discrete profile is to be created, but some authorization checking is required. For other types of action, no discrete profile should exist.

Note:
1. Use of RACFIND=YES with TYPE=DEFINE is not a recommended programming interface unless NEWNAME, NEWNAMX, CHGVOL, or ADDVOL is also specified. Creation of discrete profiles is intended to be done either by using the RACF command processors or by using the MVS routines that create data sets.
2. If you are issuing RACROUTE REQUEST=DEFINE, TYPE=DELETE and all the following are true:
   • RACFIND=YES,
   • a discrete data set name is specified, and
   • RACF finds no discrete data set name profile for that name but does find a generic profile that matches the name

   RACF considers the deletion request successful even though no profile was deleted.
3. Automatic direction of application updates does not propagate a RACROUTE REQUEST=DEFINE request for DASD data sets when the RACFIND=NO keyword is specified.
4. Automatic direction of application updates does propagate a RACROUTE REQUEST=DEFINE request for tape data sets when the RACFIND=NO keyword is specified. Even though no DATASET class profile is updated, the TAPEVOL class profile is updated.

,RESOWN=resource owner addr

specifies the address of a field containing the resource owner’s ID. If you specify RESOWN, you must specify TYPE=DEFINE and the current RELEASE parameter. The resource owner’s ID must be a valid (RACF-defined) user ID or group name, or *NONE*. If the resource owner’s ID is specified as *NONE*, RACF performs third-party authorization checking using USERID=*NONE*. The address must point to a 2-byte field followed by the resource owner’s name.

,SECLABL=addr

specifies the address of an 8-byte, left-justified character field containing the security label.

To use this keyword, you must specify RELEASE=1.9 or a later release number.

An installation can use SECLABEL to establish an association between a specific RACF security level (SECLEVEL) and a set of (zero or more) RACF security categories (CATEGORY).
REQUEST=DEFINE

,SECLVL=addr
  specifies the address of a list of installation-defined security-level identifiers.
  Each identifier is a halfword containing a value that corresponds to an
  installation-defined security-level name.

  The identifiers must be in the range 1 through 254. Only one identifier can be
  passed in the list.

  The list must start with a fullword containing the number of entries in the list
  (currently, only 0 or 1).

,STORCLA=storage_class_addr
  specifies the address of the storage class to which the resource owner must
  have authority. The address must point to a 2-byte field followed by the
  management class name. If you specify STORCLA, you must specify
  TYPE=DEFINE, RESOWN, and RELEASE=1.8.1 or a later release number.

  When specified, RACROUTE REQUEST=DEFINE processing invokes
  REQUEST=AUTH processing to verify that the RESOWNER is authorized to
  the storage class.

,TAPELBL=STD
,TAPELBL=BLP
,TAPELBL=NL
  specifies the type of tape labeling to be done:

  STD  IBM or ANSI standard labels
  BLP  Bypass label processing
  NL   Unlabeled tapes.

  For TAPELBL=BLP, the user must have the requested authority to the profile
  ICHBLP in the general-resource class FACILITY.

  For TAPELBL=NL or BLP, the user is not allowed to protect volumes with
  volume serial numbers in the format “Lnnnnn”.

  The TAPELBL parameter is passed to the RACROUTE REQUEST=DEFINE
  installation exits.

  This parameter is primarily intended for use by data-management routines to
  indicate the label type from the LABEL keyword on the JCL statement.

  This parameter is valid for CLASS=DATASET and DSTYPE=T or
  CLASS=TAPEVOL.

,TYPE=DEFINE
,TYPE=DEFINE,NEWNAME=new_resource_name_addr
,TYPE=DEFINE,NEWNAMX=extended_new_resource_name_addr
,TYPE=ADDVOL,OLDVOL=old_vol_addr
,TYPE=CHGVOL,OLDVOL=old_vol_addr
,TYPE=DELETE
  specifies the type of action to be taken:

  Note:

  If SETROPTS ADDCREATOR is in effect when a new DATASET or general
  resource profile is defined, the profile creator's user ID is placed on the profile
  access list with ALTER authority.

  If SETROPTS NOADDCREATOR is in effect when:
**REQUEST=DEFINE**

- A new generic profile is defined, the profile creator's user ID is not placed on the profile's access list. If you use profile modeling when defining a generic profile, RACF copies the access list exactly. If the creator's user ID appeared in the model's access list, the same authority is copied to the new profile.

- A new discrete DATASET or TAPEVOL profile is defined, the profile creator's user ID is placed on the profile's access list with ALTER authority. If you use profile modeling when defining one of these profiles, if the creator's user ID appeared in the model's access list, the authority is created in the new profile with ALTER authority.

- Any other new discrete profile is defined, the profiles creator's user ID is not placed on the access list. If you use profile modeling when defining one of these profiles, RACF copies the access list exactly. If the creator's user ID appeared in the model's access list, the same authority is copied to the new profile.

**DEFINE**

adds the profile of the resource to the RACF database and establishes the current user as the owner of the profile.

**DEFINE,NEWNAME**

The address points to a field containing the new name for the resource that is to be renamed. The field should be 44 bytes when class is DATASET or the maximum name length allowed for the general-resource class.

NEWNAME is valid with CLASS=DATASET, FILE, and DIRECTRY. NEWNAME is not valid with DSTYPE=T.

**DEFINE,NEWNAMX**

The address points to a structure that consists of two 2-byte length fields, followed by the entity name.

- The first 2-byte field specifies a buffer length that can be from 0 to 255 bytes. This length field refers to the length of the buffer that contains the entity name; it does not include the length of either length field.

- The second 2-byte field specifies the actual length of the entity name. This length field includes the length of the actual name without any trailing blanks; it does not include the length of either length field.

These two length fields can be used in several different ways:

- If you know the length of the entity name, you can specify 0 in the first field and the length of the entity name in the second field. When you specify the second field, note that each byte counts. This means that the entity name you specify is added to the RACF database using the specified length.

- If you choose to place the entity name in a buffer area, specify the length of the buffer in the first field. For the second field, do one of the following:
  - If you know the length of the entity name, specify the length in the second field. The length of the first field can be from 0 to 255, but must be equal to or greater than the length of the second field.
  - If you do not know the length of the entity name, specify 0 in the second field, and have RACF determine the number of characters in the entity name.
REQUEST=DEFINE

NEWNAMX is valid with CLASS=DATASET, FILE, and DIRECTRY.
NEWNAMX is not valid with DSTYPE=T.

To use this keyword, you must specify RELEASE=1.9 or later.

Guideline: Use NEWNAMX rather than NEWNAME. With
NEWNAME, the entity name you pass to RACF must be in a buffer,
the size of which is determined by the length in the class descriptor
table. If the maximum length of a class descriptor entity increases in
the future, you must modify your program to use a larger buffer. By
using NEWNAMX, you avoid this possible problem, because you
remove the class descriptor table dependency from your program.

The following parameters are ignored if you specify NEWNAME or
NEWNAMX: ACCLVL, AUDIT, CATEGORY, DATA, ERASE, EXPDT,
EXPDTX, FILESEQ, INSTLN, LEVEL, MCLASS, MENTITY, MENTX,
MGENER, MVOLSER, NOTIFY, OWNER, RETPD, SECLABL, SECLVL,
TAPELBL, UACC, UNIT, and WARNING.

ADDVOL

Adds the new volume to the definition of the specified resource.

For the DATASET class, the OLDVOL address specifies a previous volume
of a multivolume data set.

For the TAPEVOL class, the ENTITY or ENTITYX address specifies a
previous volume of a tape-volume set.

This parameter applies only to discrete profiles.

CHGVOL

Changes the volume serial number in the definition of the specified
resource from the old volume serial number identified in OLDVOL to the
new volume serial number identified in the VOLSER parameter.

This parameter applies only to discrete profiles. TYPE=CHGVOL is not
valid with DSTYPE=T.

DELETE

Removes the profile from the RACF database. (For a multivolume data set
or a tape-volume set, only the specified volume is removed from the
definition.)

If DSTYPE=T is specified, the data sets must be deleted in reverse of the
order in which they were created. For example, if file1 has data-set1, file2
has data-set2, and file3 has data-set3, you must do the RACROUTE
REQUEST=DEFINE,TYPE=DELETE,DSTYPE=T for file3, file2, and file1, in
that order.

,UACC=ALTER
,UACC=CONTROL
,UACC=UPDATE
,UACC=READ
,UACC=EXECUTE
,UACC=NONE
,UACC=reg

specifies a universal access authority for the profile. UACC must contain a
valid access authority (EXECUTE, ALTER, CONTROL, UPDATE, READ, or
NONE).

If a register is specified, the low-order byte must contain one of the following
valid access authorities:
REQUEST=DEFINE

| X'80' | ALTER |
| X'40' | CONTROL |
| X'20' | UPDATE |
| X'10' | READ |
| X'01' | NONE |
| X'09' | EXECUTE |

UACC is valid if TYPE=DEFINE is specified.

UNIT=unit addr

specifies the address of a field containing unit information. If a unit address is specified, the unit information in the data set profile is replaced by the unit information pointed to by this unit address. The unit address must point to a field containing a 1-byte length field (whose value can range from 4 through 8) followed by the actual unit information. If the value in the length field is 4, the unit information is assumed to contain a copy of the information in the UCBTYP field of the UCB. Otherwise the unit information is assumed to be in the generic form (for example, 3330-1).

UNIT is valid if TYPE=CHGVOL or TYPE=DEFINE is specified and is ignored for generic names.

WARNING=YES

WARNING=NO

If WARNING=YES is specified, the WARNING indicator is set in the profile. Access is granted to the resource and the event is logged as a warning if either the SUCCESS or FAILURES logging is requested.

This keyword is designed primarily for use by RACF commands.

WARNING is valid if TYPE=DEFINE is specified.

MF=S

specifies the standard form of the RACROUTE REQUEST=DEFINE macro instruction.

Return codes and reason codes

When you execute the macro, space for the RACF return code and reason code is reserved in the first two words of the RACROUTE parameter list. You can access them, using the ICHSAFP mapping macro, by loading the ICHSAFP pointer with the label that you specified on the list form of the macro. When control is returned, register 15 contains the SAF return code.

Note:

All return and reason codes are shown in hexadecimal. Also, note that SAF return code is presented as SAF RC and RACF return code is presented as RACF RC in the following section.

SAF RC

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
</tr>
</tbody>
</table>

RACF RC

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

Chapter 3. System macros 73
RACROUTE REQUEST=DEFINE has completed successfully.

**Note:** For TYPE=DEFINE or TYPE=DELETE, it is possible that no profile was actually added or deleted although RACF has performed all request processing. This depends on the keywords specified and the generic profiles that might exist in your system.

**Reason Code**

**Meaning**

00 Indicates a normal completion.

08 Indicates that MODEL was specified, but the SECLABEL value has not been copied because of one of two reasons:

- SETROPTS SECLABELCONTROL is on, but issuer is not SPECIAL, or
- SETROPTS MLSTABLE is on, but SETROPTS MLQUIET is not

04 The requested function could not be performed.

**RACF RC**

**Meaning**

00 No security decision could be made.

**Reason Code**

**Meaning**

00 RACF was not called to process the request because one of the following occurred:

- RACF is not installed.
- The combination of class, REQSTOR, and SUBSYS was found in the RACF router table, and ACTION=NONE was specified.
- The RACROUTE issuer specified DECOUPL=YES and a RELEASE= keyword with a higher release than is supported by this level of z/OS.

04 Indicates RACFIND=NO was specified and no generic profile applying to the data set was found.

04 RACROUTE REQUEST=DEFINE has completed processing.

**Reason Code**

**Meaning**

00 Indicates the following:

- For TYPE=DEFINE, the resource name was previously defined.
- For TYPE=DEFINE,NEWNAME or NEWNAMX, the new resource name was previously defined.
- For TYPE=DELETE, the resource name was not previously defined.

04 Indicates for TYPE=DEFINE that the data set name was previously defined on a different volume and that the option disallowing duplicate data sets was specified at IPL.

08 The requested function failed.
### REQUEST=DEFINE

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>08</td>
<td>RACROUTE REQUEST=DEFINE has completed processing.</td>
</tr>
</tbody>
</table>

#### Reason Code

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>04</td>
</tr>
<tr>
<td>08</td>
</tr>
<tr>
<td>0C</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>18</td>
</tr>
<tr>
<td>1C</td>
</tr>
<tr>
<td>20</td>
</tr>
<tr>
<td>24</td>
</tr>
<tr>
<td>28</td>
</tr>
<tr>
<td>2C</td>
</tr>
</tbody>
</table>
REQUEST=DEFINE

For TAPE data set, the USER SECLABEL does not dominate the TAPE's security label when TYPE=DELETE. When type is DEFINE, ADDVOL, OR CHGVOL, the USER security label is not equal to the TAPE's security label.

For TYPE=DEFINE, RACF has denied the authorization to allocate the data set with that name because of one of the following:
- The profile protecting it has no security label.
- The user does not dominate the security label of the profile protecting the entity.
- The data set is not protected by any profile.

For TYPE=DEFINE, NEWNAME= or NEWNAMX=, RACF has denied the authorization to rename the data set because the new data set name is either:
- Protected by a profile with no security label.
- Protected by a profile whose security label the user does not dominate.
- Protected by no profile.

The request to rename a profile is denied because the SETROPTS MLS option is in effect and one of the following is true:
- The entity is not protected by a profile.
- The entity is protected by a profile with no security label.
- The entity is protected by a profile whose security label the user cannot possibly dominate.

The request to rename the resource is denied because the SETROPTS MLS option is in effect and one of the following is true:
- The new name is not protected by any profile.
- The new name is protected by a profile with no security label.
- The user can never equal the security label that protects the new name.

The request to rename the resource is denied because the security label of the generic profile protecting the new data set name does not dominate the security label of the generic profile protecting the entity name. This is equivalent to writing down, and is disallowed because the SETROPTS MLS option is in effect.

The request to RACROUTE REQUEST=DEFINE is denied because the profile defined has a security label different from the generic profile covering it and that the SETROPTS MLSTABLE option is in effect. This happens under the following circumstances:
- The request to define a DASDVOL data set is denied because the parent generic profile has a different security label.
REQUEST=DEFINE

- The request to define a TAPEVOL data set failed for one of the following reasons:
  - The data set has a parent generic with a different security label.
  - The tape volume is not defined and a generic tape profile exists with a security label different from the security label added to the discrete tape profile.
  - The tape volume is defined and the security label being added is different from the security label of the generic profile protecting it.
- The request to add a volume is denied because the new volume has a security label different from the security label of the generic profile protecting it.
- The request to rename the profile is denied because the security label of the generic profile covering the new name is different from the security label of the entity profile.

48 The request to REQUEST=DEFINE is failed because the user is not SPECIAL, SETROPTS GENERICOWNER is in effect, and one of the following happens:
- For TYPE=DEFINE, the user cannot define the profile because of generic owner requirements with respect to the generic profile covering the entity name. This restriction does not apply to data sets.
- For TYPE=DEFINE, NEWNAME= or NEWNAMX=, the user does not meet the generic owner requirements with respect to the less specific generic profile for NEWNAME. This reason does not apply to the DATASET class.
- For TYPE=ADDVOL, the user is not allowed to add a volume profile because a generic profile exists in the class and the user does not meet the generic owner requirement.
- For TYPE=DEFINE, DSTYPE=T, and CLASS=DATASET, the request is failed because a discrete, automatic, tape profile is defined, but the user does not meet the generic owner requirement with respect to the generic TAPEVOL profile.

4C Indicates that the RESOWNER is a revoked user ID.

0C For TYPE=DEFINE,NEWNAME or NEWNAMX, the old resource name was not defined; or for CLASS=DATASET, if the generation-data-group (GDG) modeling function is active, an attempt was made to rename a GDG name to a name that requires the creation of a new profile; or if generic profile checking is active, the old resource name was protected by a generic profile and there is no generic profile that protects the new resource name. This last case refers only to an attempt to rename an existing profile, which cannot be found.

10 For TYPE=DEFINE with MENTITY or MENTX, the model resource was not defined.

64 Indicates that the CHECK subparameter of the RELEASE keyword was
REQUEST=DEFINE

specified on the execute form of the RACROUTE REQUEST=DEFINE macro; however, the list form of the macro does not have the same RELEASE parameter. Macro processing terminates.

Example 1
Invoke RACF to define a discrete profile for a non-VSAM data set residing on the volume pointed to by register 8. Register 7 points to the data set name. All successful requests for update authority to the data set are to be audited, as well as all unsuccessful ones.

```
RACROUTE REQUEST=DEFINE,ENTITY=(R7),VOLSER=(R8),CLASS='DATASET', X
  AUDIT=(SUCCESS(UPDATE),FAILURES),WORKA=RACWK,    X
  RACFIND=YES
```

```
RACWK DS CL512
```

Example 2
Use the standard form of the RACROUTE REQUEST=DEFINE macro to define a discrete data set profile for a non-VSAM DASD data set. The data set for which you are creating a profile is a non-VSAM DASD data set named DSNAME. It resides on a volume named VOLID. You want to create a discrete profile by specifying the RACFIND keyword. In addition, you want to notify the user called USERNAME of any access attempts that have been rejected because they exceed the UACC you are allowing for READ.

```
RACROUTE REQUEST=DEFINE,ENTITY=DSNAME,VOLSER=VOLID, X
  CLASS='DATASET',UACC=READ,WORKA=RACWK,    X
  RACFIND=YES,NOTIFY=USERNAME,RELEASE=1.7
```

```
RACWK DS CL512
```

Example 3
Use the standard form of the macro to check the authority of a user to define a discrete data set profile for a non-VSAM DASD data set, but do not actually build the profile. The name of the data set is DSNAME.

```
RACROUTE REQUEST=DEFINE,ENTITY=DSNAME,VOLSER=VOLID, X
  CLASS='DATASET',RACFIND=NO,WORKA=RACWK
```

```
RACWK DS CL512
```

Example 4
Use the standard form of the macro to define a generic data set profile named PROFNAME. As a model for the new profile, use the discrete profile named MDELPROF whose volume serial number is in MDELVOL. Notify the user named USERNAME of any access attempts that have been rejected because they exceed the UACC you are allowing for READ.

```
RACROUTE REQUEST=DEFINE,ENTITY=PROFNAME, X
  CLASS='DATASET',GENERIC=YES,MENTITY=MDELPROF, X
  MVOLSER=MDELVOL,UACC=READ,WORKA=RACWK,    X
  NOTIFY=USERNAME,RELEASE=1.7
```

```
RACWK DS CL512
```

Example 5
Use the standard form of the macro to define a tape-volume profile for a volume whose ID is VOLID. Allow a universal access level of READ.
REQUEST=DEFINE

RACROUTE REQUEST=DEFINE,ENTITY=VOLID,CLASS='TAPEVOL', UACC=READ,WORKA=RACWK

: RACWK DS CL512

Example 6
Use the standard form of the macro to delete a discrete data set profile named DSNAME located on the volume named VOLID.
RACROUTE REQUEST=DEFINE,TYPE=DELETE,ENTITY=DSNAME, VOLSER=VOLID,CLASS='DATASET',WORKA=RACWK

: RACWK DS CL512

RACROUTE REQUEST=DEFINE (list form)
The list form of the RACROUTE REQUEST=DEFINE macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=DEFINE macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

| name | 
| name: Symbol. Begin name in column 1. | 
| b | One or more blanks must precede RACROUTE. | 
| RACROUTE | 
| b | One or more blanks must follow RACROUTE. | 

REQUEST=DEFINE

,ACCLVL=access value addr
,ACCLVL=(access value addr,parm list addr)
,ACEE=acee addr
,AUDIT=NONE
,AUDIT=audit value
,AUDIT=(audit value (access level), audit value (access level))
,CHKAUTH=YES,CHKAUTH=NO
,CLASS=class name
,CLASS=class name addr
,DATA=data addr
,DSTYPE=N

Default: AUDIT=READ
Default: CHKAUTH=NO
Default: CLASS=DATASET
Default: DSTYPE=N
REQUEST=DEFINE

...DSTYPE=V
..DSTYPE=M
...DSTYPE=T

...ENTITY=profile name addr
..ENTITYX=extended profile name addr

Note: ENTITY or ENTITYX must be specified on the list, execute, or modify form of the macro.

...ENVIR=VERIFY
.. Specifies that only verification is to be done.
.. Default: Normal RACROUTE REQUEST=DEFINE processing.

...ERASE=YES
.. Default: ERASE=NO

...EXPDT=expire-date addr
...EXPDTX=extended expire-date addr
...RETPD=retx-period addr
.. Default: LEVEL=zero.

...FILESEQ=number
.. number: 1–65535

...GENERIC=YES
.. Default: GENERIC=ASIS

...INSTLN=parm list addr
.. parm list addr: A-type address

...LEVEL=number
.. Default: LEVEL=zero.

...MCLASS='class name'
...MCLASS=class name addr
.. Default: MCLASS=DATASET

...MENTITY=entity addr
...MENTX=extended entity addr

...MGENER=ASIS
.. Default: MGENER=ASIS

...MGMTCLA=management type addr
.. Default: See description of parameter.

...MVOLSER=volser addr
.. volser addr: A-type address

...NOTIFY=notify-id addr
.. notify-id addr: A-type address

...OWNER=owner id addr
.. owner id addr: A-type address

...RACFIND=YES
.. Default: RACFIND=NO

...SECLBL=addr
.. addr: A-type address

...SECVL=addr
.. addr: A-type address
The parameters are explained under the standard form of the RACROUTE REQUEST=DEFINE macro instruction with the following exception:

\[MF=L\]

specifies the list form of the RACROUTE REQUEST=DEFINE macro instruction.

**RACROUTE REQUEST=DEFINE (execute form)**

The execute form of the RACROUTE REQUEST=DEFINE macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=DEFINE macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.
**REQUEST=DEFINE**

- **name**: Symbol. Begin *name* in column 1.

- **b**: One or more blanks must precede RACROUTE.

**RACROUTE**

- **b**: One or more blanks must follow RACROUTE.

**REQUEST=DEFINE**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCLVL=access value addr, ACCLVL=(access value addr, parm list addr)</td>
<td>access value addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>ACEE=acee addr</td>
<td>acee addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>AUDIT=None/AUDIT=audit value/AUDIT=(audit value (access level), audit value (access level))/AUDIT=reg</td>
<td>audit value: ALL, SUCCESS, or FAILURES</td>
</tr>
<tr>
<td>CHKAUTH=YES/CHKAUTH=NO</td>
<td></td>
</tr>
<tr>
<td>CLASS=class name addr</td>
<td>class name addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>DATA=data addr</td>
<td>data addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>DSTYPE=N/DSTYPE=V/DSTYPE=M/DSTYPE=T</td>
<td></td>
</tr>
<tr>
<td>ENTITY=profile name addr/ENTITYX=extended profile name addr</td>
<td>profile name addr: Rx-type address</td>
</tr>
<tr>
<td>ENTITYX=extended profile name addr</td>
<td>extended profile name addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>ENVIR=VERIFY</td>
<td>Specifies that only verification is to be done.</td>
</tr>
<tr>
<td>ERASE=YES/ERASE=NO</td>
<td></td>
</tr>
<tr>
<td>EXPDT=expir-date addr/EXPDTX=extended expir-date addr</td>
<td>expir-date addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>RETPD=retm-period addr</td>
<td>retm-period addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>FILESEQ=number/FILESEQ=reg</td>
<td>number: 1–65535</td>
</tr>
<tr>
<td>GENERIC=YES/Generic=ASIS</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** ENTITY or ENTITYX must be specified on the list, execute, or modify form of the macro.
REQUEST=DEFINE

,INSTLN=parm list addr
  parm list addr: Rx-type address or register (2) – (12)

,LEVEL=number
,LEVEL=reg
  reg: Register (2) – (12)

,MCLASS=class
  class name addr: Rx-type address or register (2) – (12)

,MENTITY=entity addr
  entity addr: Rx-type address or register (2) – (12)

,MENTITY=extended
  extended entity addr: Rx-type address or register (2) – (12)

,MENTITY=extended
  extended entity addr: Rx-type address or register (2) – (12)

,MGENER=ASIS
,MGENER=YES

,MGENER=ASIS
,MGENER=YES

,GMTCLASS=management
  management type addr: Rx-type address or register (2) – (12)

,MVOLSER=volser addr
  volser addr: Rx-type address or register (2) – (12)

,NOTIFY=notify-id addr
  notify-id addr: Rx-type address or register (2) – (12)

,OWNER=owner id addr
  owner id addr: Rx-type address or register (2) – (12)

,RACFIND=YES
,RACFIND=NO

,RACFIND=YES
,RACFIND=NO

,RELEASE=number
  number: See Standard Form

,RELEASE=(,CHECK)
  Default: RELEASE=1.6

,RELEASE=(,CHECK)
  Default: RELEASE=1.6

,RESOWN=resource
  resource owner addr: Rx-type address or register (2) – (12)

,RESOWN=resource
  resource owner addr: Rx-type address or register (2) – (12)

,SECLABL=addr
  addr: Rx-type address or register (2) – (12)

,SECLVL=addr
  addr: Rx-type address or register (2) – (12)

,STORCLA=storage class
  storage class addr: Rx-type address or register (2) – (12)

,TAPFSLB=STD
,TAPFSLB=BLP
,TAPFSLB=NL

,TAPFSLB=STD
,TAPFSLB=BLP
,TAPFSLB=NL

,TYPE=DEFINE
,TYPE=DEFINE,
  new resource name addr: A-type address or register (2) – (12)

,TYPE=DEFINE,
  new resource name addr: A-type address or register (2) – (12)

,TYPE=DEFINE,
  new resource name addr: A-type address or register (2) – (12)

,TYPE=ADDVOL,
  old vol addr: A-type address or register (2) – (12)
REQUEST=DEFINE

vol addr
 ,TYPE=CHGVOL
OLDVOL=old
vol addr
 ,TYPE=DELETE

,UACC=ALTER
,UACC=CONTROL
,UACC=UPDATE
,UACC=READ
,UACC=EXECUTE
,UACC=NONE

,UACC=reg

,UNIT=unit addr

,VOLSER=vol addr

Note: VOLSER is required for CLASS=DATASET and DSTYPE not equal to M when a discrete profile name is used.

,WARNING=YES
,WARNING=NO

,MF=(E,ctrl addr)

The parameters are explained under the standard form of the RACROUTE REQUEST=DEFINE macro instruction with the following exceptions:

,MF=(E,ctrl addr)

specifies the execute form of the RACROUTE REQUEST=DEFINE macro using a remote, control-program parameter list.

,RELEASE=number
,RELEASE=(,CHECK)

specifies the RACF release level of the parameter list to be generated by this macro.

To use the parameters associated with a release, you must specify the release number of that release or a later release number. If you specify an earlier release level, the parameter is not accepted by macro processing, and an error message is issued at assembly time.

The default is RELEASE=1.6.

When you specify the RELEASE keyword, checking is done at assembly time. Compatibility between the list and execute forms of the RACROUTE REQUEST=DEFINE macro is validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.
The modify form of the RACROUTE REQUEST=DEFINE macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=DEFINE macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

```
ITEM       DESCRIPTION
name       name: Symbol. Begin name in column 1.
            b One or more blanks must precede RACROUTE.
RACROUTE   One or more blanks must follow RACROUTE.
REQUEST=DEFINE
            ,ACCLVL=access value addr: Rx-type address or register (2) – (12)
            ,ACCLVL=(access value addr, parm list addr): Rx-type address or register (2) – (12)
            ,ACEE=acee addr: Rx-type address or register (2) – (12)
            ,AUDIT=NONE
            ,AUDIT=audit value: audit value: ALL, SUCCESS, or FAILURES
            ,AUDIT=(audit value (access level), audit value (access level)): access level: READ, UPDATE, CONTROL, or ALTER
            ,AUDIT=reg: Register (2) – (12)
            ,CHKAUTH=YES
            ,CHKAUTH=NO
            ,CLASS=class name addr: Rx-type address or register (2) – (12)
            ,DATA=data addr: Rx-type address or register (2) – (12)
            ,DSTYPE=N
            ,DSTYPE=V
            ,DSTYPE=M
            ,DSTYPE=T
            ,ENTITY=profile name addr: Rx-type address
            ,ENTITYX=extended profile name addr: Rx-type address or register (2) – (12)
Note: ENTITY or ENTITYX must be specified on either the list, execute, or modify form of the macro.
            ,ENVIR=VERIFY Specifies that only verification is to be done.
            ,ERASE=YES
            ,ERASE=NO
            ,EXPDT=expir-date addr: Rx-type address or register (2) – (12)
```

Note: ENTITY or ENTITYX must be specified on either the list, execute, or modify form of the macro.
REQUEST=DEFINE

,EXPDTX=extended expir-date addr
,RETPD=ret-period addr
,FILESEQ=number
,FILESEQ=reg
,Generic=YES
,Generic=ASIS
,INSTLN=parm list addr
,LEVEL=number
,LEVEL=reg
,MCLASS=class name addr
,MENTITY=entity addr
,MENTITYX=extended entity addr
,GENER=ASIS
,MGENER=YES
,MGMTCLA=management type addr
,MVOLSER=volser addr
,NOTIFY=notify-id addr
,OWNER=owner id addr
,RACFIND=YES
,RACFIND=NO
,RESOWN=resource owner addr
,SECLABL=addr
,SECLVL=addr
,STORCLA=storage class addr
,TAPELBL=STD
,TAPELBL=BLP
,TAPELBL=NL
,TYPE=DEFINE
,NEWNAME=new resource name addr

extended expir-date addr: Rx-type address or register (2) – (12)
ret-period addr: Rx-type address or register (2) – (12)
number: 1–65535
reg: Register (2) – (12)
entity addr: Rx-type address or register (2) – (12)
extended entity addr: Rx-type address or register (2) – (12)
management type addr: Rx-type address or register (2) – (12)
volser addr: Rx-type address or register (2) – (12)
notify-id addr: Rx-type address or register (2) – (12)
owner id addr: Rx-type address or register (2) – (12)
resource owner addr: Rx-type address or register (2) – (12)
addr: Rx-type address or register (2) – (12)
addr: Rx-type address or register (2) – (12)
storage class addr: Rx-type address or register (2) – (12)
new resource name addr: A-type address or register (2) – (12)
REQUEST=DEFINE

,TYPE=DEFINE, extended new resource name addr: A-type address or register (2) – (12)
NEWNAMX=extended new resource name addr
,TYPE=ADDVOL, old vol addr: A-type address or register (2) – (12)
OLDVOL=old vol addr
,TYPE=CHGVOL, old vol addr: A-type address or register (2) – (12)
OLDVOL=old vol addr
,TYPE=DELETE
,UACC=ALTER
,UACC=CONTROL
,UACC=UPDATE
,UACC=READ
,UACC=EXECUTE
,UACC=None
,UACC=reg reg: Register (2) – (12)
,UNIT=unit addr unit addr: Rx-type address or register (2) – (12)
,VOLSER=vol addr vol addr: Rx-type address or register (2) – (12)

Note: VOLSER is required for CLASS=DATASET and DSTYPE not equal to M when a discrete profile name is used.

,WARNING=YES Note: Warning is valid if TYPE=DEFINE is specified.
,WARNING=NO
,MF=(M,ctrl addr) ctrl addr: Rx-type address or register (1) or (2) – (12)

The parameters are explained under the standard form of the RACROUTE REQUEST=DEFINE macro instruction with the following exceptions:

,MF=(M,ctrl addr)
specifies the modify form of the RACROUTE REQUEST=DEFINE macro using a remote control-program parameter list.

RACROUTE REQUEST=DIRAUTH: Directed authorization check of security classification

This service compares two security labels. The security labels may be passed directly, or as part of an ACEE or UTOKEN. The class name determines the type of comparison made between the security labels, unless the TYPE parameter is specified. Note that the security labels should be obtained from the same system since a SECLABEL name may not represent the same security classification on different systems. Components that need to compare security labels for equality may do a check for an exact match without issuing this macro.

The message-transmission managers (that is, VTAM®, TSO/E, and Session Manager) use the RACROUTE REQUEST=DIRAUTH macro with RTEEN specified to ensure that the receiver of a message, represented by the ACEE of the current address space, meets security-label authorization requirements. That is, the security label of the receiver of the message must dominate (be equal to or higher than) the security label of the message. When invoked as these managers do, with just the RTEEN or just the RTEEN and LOG keywords specified, if the security
REQUEST=DIRAUTH

label of the receiver does not dominate the security label of the message, DIGAUTH performs additional processing to determine if the receiver has access to any security label that could dominate the message.

To use this service, you must specify RELEASE=1.9 or a later release number.

The caller of RACROUTE REQUEST=DIRAUTH must be authorized (APF-authorized, in system key 0–7, or in supervisor state).

The caller cannot hold any locks when issuing RACROUTE REQUEST=DIRAUTH.

This request is SRB-mode compatible. When issuing RACROUTE REQUEST=DIRAUTH in SRB mode, you must ensure that the jobstep task pointed to by the ASCBXTCB field in the target address space is active when you schedule the SRB and that it remains active until the SRB completes.

Note: In order to ensure that the SRB does not run after the ASCBXTCB task completes, you must enable purgeDQ to deal with that SRB. In particular:

- If using SCHEDULE, SRBPASID must be set to the ASID, and SRBPTCB must be set to the contents of ASCBXTCB.
- If using IEAMSCMD, PURGESTOKEN must be specified with the STOKEN of the space, and PTCBADDR must be specified with the contents of ASCBXTCB.

When the task terminates, the purgeDQ issued causes the SRB's resource manager termination routine (RMTR) to be driven if the SRB has not begun to run, or waits for the SRB to complete if the SRB has begun to run.

RACROUTE REQUEST=DIRAUTH can also be invoked from a cross memory environment, when in task or SRB mode. The ACEE used for authorization checking must reside in the HOME address space (unless ACEEALET specifies a different address space). Callers in cross-memory mode must be in supervisor state. RACROUTE REQUEST=DIRAUTH interrogates the setting of SETR MLS during dominance checking for ACCESS=READWRITE and ACCESS=WRITE to determine if write-down is allowed on the system. It does not support WARNING mode for SETR MLS and will process write-down violations as failures.

RACROUTE REQUEST=DIRAUTH does not check whether or not security labels are required, nor does it always bypass security label checking if the ACEE indicates trusted or privileged authority. If a full authorization check, including the checking of security labels, is needed then RACROUTE REQUEST=AUTH, RACROUTE REQUEST=FASTAUTH, or the ck_access callable service should be used.

RACROUTE REQUEST=DIRAUTH (standard form)

The standard form of the RACROUTE REQUEST=DIRAUTH macro is written as follows. For a description of additional keywords that you can code and additional parameters that are required on the RACROUTE request, but that are not specific to this request type, see "RACROUTE (standard form)" on page 13.
REQUEST=DIRAUTH

Note:

RACROUTE REQUEST=DIRAUTH requires an ACEE under many circumstances. For most applications, the system will have created an ACEE to represent the active user. However, for special cases where no ACEE exists, the invoker must create one before invoking RACROUTE REQUEST=DIRAUTH.

The most common way to create an ACEE is to issue a RACROUTE REQUEST=VERIFY, specifying ENVIR=CREATE. After all RACROUTE invocations are complete, the invoker should issue RACROUTE REQUEST=VERIFY with ENVIR=DELETE specified, to delete the ACEE previously created.

The parameters are explained as follows:

,ACCESS=READ
,ACCESS=READWRITE
,ACCESS=WRITE

,ACEE=ACEE addr
,USERSECLABEL=seclabel addr

,CLASS='class name'
,CLASS=class name addr

,LOG=ASIS
,LOG=NOFAIL
,LOG=NONE

,LOGSTR=logstr addr

,TYPE=MAC
,TYPE=EQUALMAC
,TYPE=RVRSMAC

,MF=S

Note:

REQUEST=DIRAUTH requires an ACEE under many circumstances. For most applications, the system will have created an ACEE to represent the active user. However, for special cases where no ACEE exists, the invoker must create one before invoking RACROUTE REQUEST=DIRAUTH.

The most common way to create an ACEE is to issue a RACROUTE REQUEST=VERIFY, specifying ENVIR=CREATE. After all RACROUTE invocations are complete, the invoker should issue RACROUTE REQUEST=VERIFY with ENVIR=DELETE specified, to delete the ACEE previously created.
REQUEST=DIRAUTH

,ACCESS=WRITE
specifies the type of access attempt the user is requesting. This value is used along with TYPE to determine how the user and resource security labels are evaluated. If ACCESS is not specified, READ access will be assumed. When specifying this parameter, RELEASE=7708 or later must also be specified.

,ACEE=ACEE addr
specifies the address of the user's ACEE. If specified, the security label and write-down indicator from this ACEE will be used to determine authorization. In cross-memory mode, the ACEE must be in the home address space unless the ACEEALET= keyword specifies another address space. If neither ACEE nor USERSECLABEL is specified, the ACEE of the address space (the home address space in cross memory mode) will be used. When both ACEE and USERSECLABEL are specified, ACEE is ignored. When specifying this parameter, RELEASE=7708 or later must also be specified.

,ACEEALET=alet addr
specifies the address of the 4-byte ALET to be used to access the ACEE specified on the ACEE= keyword. If the ACEE= keyword is not specified, the ACEEALET= keyword is ignored. When you use A-type or RX-type notation, alet addr specifies the name of a 4-byte field that contains the ALET. When you use register notation, alet addr specifies a register that contains the address of a 4-byte field that contains the ALET. If this keyword is not specified, the ACEE is located as defined in the ACEE= keyword description.

Keyword requirements:
1. Run in supervisor state or system key (0–7).
2. Ensure that the address space identified by ACEEALET= is marked non-swappable.
3. Ensure the specified ALET represents a valid entry in the DU-AL of the work unit or the PASN-AL of the current primary address space.
4. Specify RELEASE=7708 or later.

,CLASS='class name'
,CLASS=class name addr
specifies a 1–8 character class name, or the address of an 8-byte area containing the class name, left-justified and padded with blanks. If a class name is specified and TYPE= is not specified, the CDT entry of the class is used to determine the type of access check. When specifying this parameter, RELEASE=7708 or later must also be specified.

,LOG=ASIS
,LOG=NOFAIL
,LOG=NONE
specifies the type of access attempts that RACF is to record on the SMF data set. Failures that result in SMF recording will also produce an ICH408I message unless MSGSUPP=YES is specified.

ASIS RACF records the event in the manner specified on the SETR LOGOPTIONS command for the DIRAUTH resource class, unless a class name is specified. If a class name is specified, the SETR LOGOPTIONS for that class will determine whether or not auditing will occur.

NOFAIL
If the authorization check fails, RACF does not record the attempt.
If the authorization check succeeds, RACF records the attempt as it does in ASIS.
REQUEST=DIRAUTH

NONE

RACF does not record the event or issue any messages.

\,LOGSTR=logstr addr

specifies the address of a 1-byte length field followed by character data to be written to the system-management-facilities (SMF) data set together with any RACF audit information, if logged.

\,RESCSECLABEL=seclabel addr

specifies the address of an 8-byte area containing the security label of a resource. Resources include server address spaces, network resources, files, data sets, etc. The security label must be specified in uppercase, left-justified, and padded with blanks. If RESCSECLABEL is not specified, then RTOKEN must be specified. If both RESCSECLABEL and RTOKEN are specified, RTOKEN is ignored. When specifying this parameter, RELEASE=7708 or later must also be specified.

\,RTOKEN=message token addr

specifies the address of the token of a resource (RTOKEN). The RTOKEN data contains the user token (UTOKEN) of the creator of the resource. If the first two bytes (length and version) are equal to 0, it is the same as not specifying the RTOKEN. If RTOKEN is not specified, RESCSECLABEL must be specified. If both RESCSECLABEL and RTOKEN are specified, RTOKEN is ignored. When RTOKEN is specified with no additional keywords except the optional LOG keyword, then the DIRAUTH class must be active.

\,TYPE=MAC
\,TYPE=EQUALMAC
\,TYPE=RVRSMAC

specifies the type of dominance checking to be done for the security labels specified: normal mandatory access (MAC) checking, equal MAC checking and reverse MAC checking. If TYPE is not specified and CLASS is specified, then CLASS name is used to determine the type of checking to be done. If neither TYPE nor CLASS is specified, normal MAC checking will be performed. When specifying this parameter, RELEASE=7708 or later must also be specified.

The type of dominance checking performed depends not only on the TYPE value, but also the ACCESS value, the SETROPTS MLS or NOMLS setting, and the user’s write-down mode (as set by permission to the FACILITY class profile IRR.WRITEDOWN.BYUSER). Table 4 describes the security label dominance required for successful authorization checking with each MAC type and ACCESS value.

Table 4. Type of security label dominance required for successful authorization checking with each MAC type and ACCESS value

<table>
<thead>
<tr>
<th>Values for the ACCESS parameter</th>
<th>SETROPTS NOMLS in effect or the user in write-down mode</th>
<th>SETROPTS MLS in effect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TYPE=MAC</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>READ</td>
<td>User dominance</td>
<td></td>
</tr>
<tr>
<td>READ/WRITE</td>
<td>User dominance</td>
<td>Equivalence</td>
</tr>
<tr>
<td>WRITE</td>
<td>User or resource dominance</td>
<td>Resource dominance</td>
</tr>
<tr>
<td><strong>TYPE=EQUALMAC</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>READ</td>
<td></td>
<td>Equivalence</td>
</tr>
<tr>
<td>READ/WRITE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WRITE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 4. Type of security label dominance required for successful authorization checking with each MAC type and ACCESS value (continued)

<table>
<thead>
<tr>
<th>Values for the ACCESS parameter</th>
<th>SETROPTS NOMLS in effect or the user in write-down mode</th>
<th>SETROPTS MLS in effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE=RVRSMAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>READ</td>
<td>Resource dominance</td>
<td></td>
</tr>
<tr>
<td>READ/WRITE</td>
<td>Resource dominance</td>
<td>Equivalence</td>
</tr>
<tr>
<td>WRITE</td>
<td>User or resource dominance</td>
<td>User dominance</td>
</tr>
</tbody>
</table>

,,USERSECLABEL=seclabel addr

specifies the address of an 8-byte area containing the security label of the user. The security label must be specified in uppercase, left-justified, and padded with blanks. If neither USERSECLABEL nor ACEE is specified, the ACEE of the address space will be used. If both USERSECLABEL and ACEE are specified, ACEE is ignored. The write-down privilege will be ignored if USERSECLABEL is used. When specifying this parameter, RELEASE=7708 or later must also be specified.

The dominance checking performed depends not only on the TYPE keyword, but also the MAC keyword, the SETROPTS MLS/NOMLS setting, and whether write-down mode is in effect (as set by permission to the facility class profile IRR.WRITEDOWN.BYUSER). The above description applies to TYPE=READ, for which the other factors have no affect. The following table describes the dominance checking for all MAC specifications. (Add table here.)

,,MF=S

specifies the standard form of the RACROUTE REQUEST=DIRAUTH macro instruction.

Return codes and reason codes

When you execute the macro, space for the RACF return code and reason code is reserved in the first two words of the RACROUTE parameter list. You can access them using the ICHSAFP mapping macro, by loading the ICHSAFP pointer with the label that you specified on the list form of the macro. When control is returned, register 15 contains the SAF return code.

<table>
<thead>
<tr>
<th>SAF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACROUTE REQUEST=DIRAUTH has completed successfully.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Authorized to the resource</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Function completed successfully.</td>
</tr>
<tr>
<td>04</td>
<td>RTOKEN passed belongs to an operator or a trusted user.</td>
</tr>
</tbody>
</table>
REQUEST=DIRAUTH

This code is returned only if keywords are limited to 
RTOKEN and, optionally, LOG.

04 The requested function could not be performed.

RACF RC

Meaning

00 No security decision could be made.

Reason Code

Meaning

00 RACF was not called to process the request because one of 
the following occurred:
   • RACF is not installed.
   • The combination of class, REQSTOR, and SUBSYS was 
     found in the RACF router table, and ACTION=NONE 
     was specified.
   • The RACROUTE issuer specified DECOUPL=YES and a 
     RELEASE= keyword with a higher release than is 
     supported by this level of z/OS.

04 DIRAUTH cannot make a decision.

Reason Code

Meaning

00 One of the following has occurred:
   • ACEE does not contain TOKEN information.
   • The DIRAUTH class or RACF is not active. In this case, 
     this return code is returned only when the RTOKEN 
     keyword is used without additional keywords (other 
     than the optional LOG keyword).

04 Reserved.

08 The definition of one of the provided security labels was 
not found.

0C The translation of one of the security labels to its defining 
security level and categories failed.

10 The SECLABEL general-resource class was either not 
activated by SETROPTS CLASSACT(SECLABEL) or not 
brought into storage by SETROPTS RACLST(SECLABEL).

14 No defining security level exists in one of the SECLABEL 
profiles.

18 Class not found and is needed to determine TYPE or 
LOGOPTIONS.

1C Could not access the RACLSTed dataspace for the 
SECLABEL class due to an ALESERV failure.

FF An unexpected error occurred while checking security-label 
authorization.

0C Parameters passed to DIRAUTH are not valid.

Reason Code

Meaning
REQUEST=DIRAUTH

00 Either the resource token (RTOKEN) or security label (RESCSECLABEL) was not specified.

04 The resource token (RTOKEN) specified is not valid.

08 Invoked in cross-memory mode but the calling program is not running in supervisor state.

0C The ACEEALET= keyword was specified but the calling program is not running in supervisor state or system key (0–7).

10 Invoked in cross-memory mode but the RTOKEN keyword was used without additional keywords (other than the optional LOG keyword).

08 The requested function failed.

RACF RC

Meaning

08 Not authorized to the resource. One of the following two cases applies, depending on the keywords used:

• When the RTOKEN keyword is used without additional keywords (other than the optional LOG keyword), the following reason codes apply:

  Reason Code
  Meaning

  00 The security label in the user’s ACEE does not currently dominate that of the RTOKEN. However, the user does possess a security label that can dominate that of the RTOKEN.

  04 The security label in the user’s ACEE does not dominate that of the RTOKEN, and the user does not possess a security label that can dominate that of the RTOKEN.

• When any additional keyword is used other than the RTOKEN keyword alone, or RTOKEN with the optional LOG keyword, the following reason code applies:

  Reason Code
  Meaning

  00 Not authorized to the resource.

64 Indicates that the CHECK subparameter of the RELEASE keyword was specified on the execute form of the RACROUTE REQUEST=DIRAUTH macro; however, the list form of the macro does not have the same RELEASE parameter. Macro processing terminates.

Example

Invoke the RACROUTE REQUEST=DIRAUTH macro on behalf of the VTAM resource manager to perform security-label authorization checking in the “receiving” user’s address space to ensure that the receiver’s security label dominates that of the message. Specify that RACF should audit the event as specified in the SETROPTS LOGOPTIONS value for the DIRAUTH class.

In this example, the receiver’s security label can never dominate the security label found in the TOKEN specified on the RTOKEN= keyword. The return code received from the DIRAUTH service is 8 and the reason code is 4.
REQUEST=DIRAUTH

Note: The message cannot be received by anyone other than the person to whom it was directed.

```
RACROUTE REQUEST=DIRAUTH,WORKA=RACWK,RTOKEN=(8), X
   LOG=ASIS,RELEASE=1.9.2
...
```

RACWK DS CL512

**RACROUTE REQUEST=DIRAUTH (list form)**

The list form of the RACROUTE REQUEST=DIRAUTH macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=DIRAUTH macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

```
name name: Symbol. Begin name in column 1.

b One or more blanks must precede RACROUTE.

RACROUTE

b One or more blanks must follow RACROUTE.
```

```
REQUEST=DIRAUTH

,RTOKEN=message token addr message token addr: A-type address
,RESCSECLABEL=seclabel addr seclabel addr: A-type address

,ACCESS=READ
,ACCESS=READWRITE
,ACCESS=WRITE

,ACEE=ACEE addr ACEE addr: A-type address
,USERSECLABEL=seclabel addr

,ACEEALET=ALET addr ALET addr: A-type address

,CLASS='class name' class name: 1–8 character name
,CLASS=class name addr

,LOG=ASIS
,LOG=NOFAIL
,LOG=NONE

,LOGSTR=logstr addr logstr addr: A-type address

,TYPE=MAC
,TYPE=EQUALMAC
,TYPE=RVRSMAC

,MF=L
```

The parameters are explained under the standard form of the RACROUTE REQUEST=DIRAUTH macro with the following exception:
REQUEST=DIRAUTH

,\texttt{MF=L}

specifies the list form of the RACROUTE REQUEST=DIRAUTH macro instruction.

**RACROUTE REQUEST=DIRAUTH (execute form)**

The execute form of the RACROUTE REQUEST=DIRAUTH macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=DIRAUTH macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
</tbody>
</table>

RACROUTE

| b            | One or more blanks must follow RACROUTE. |

REQUEST=DIRAUTH

,\texttt{RTOKEN=message token addr}

,\texttt{RESCSECLABEL=seclabel addr}

,\texttt{ACCESS=READ}

,\texttt{ACCESS=READWRITE}

,\texttt{ACEE=ACEE addr}

,\texttt{USERSECLABEL=seclabel addr}

,\texttt{ACEEALET=ALET addr}

,\texttt{CLASS=class name addr}

,\texttt{LOG=ASIS}

,\texttt{LOG=NOFAIL}

,\texttt{LOG=NONE}

,\texttt{LOGSTR=logstr addr}

,\texttt{RELEASE=number}

,\texttt{RELEASE=(,CHECK)}

,\texttt{RELEASE=(number,CHECK)}

,\texttt{RTOKEN=message token addr}

,\texttt{TYPE=MAC}

,\texttt{TYPE=EQUALMAC}

,\texttt{TYPE=RVRSMAC}

,\texttt{MF=(E,ctrl addr)}

message token addr: Rx-type address or register (2) – (12)

seclabel addr: Rx-type address or register (2) – (12)

ACEE addr: Rx-type address or register (2) – (12)

seclabel addr: Rx-type address or register (2) – (12)

ALET addr: Rx-type address or register (2) – (12)

class name addr: Rx-type address or register (2) – (12)

ASIS

Default=ASIS

typestr addr: Rx-type address or register (2) – (12)

number: See standard form

message token addr: Rx-type address or register (2) – (12)

Default=MAC

ctrl addr: Rx-type address or register (1) or (2) – (12)
The parameters are explained under the standard form of the RACROUTE REQUEST=DIRAUTH macro with the following exceptions:

- `RELEASE=number`
- `RELEASE=(, CHECK)`
- `RELEASE=(number, CHECK)`

specifies the RACF release level of the parameter list to be generated by this macro.

To use the parameters associated with a release, you must specify the release number of that release or a later release number. If you specify an earlier release level, the parameter is not accepted by macro processing, and an error message is issued at assembly time.

When you specify the `RELEASE` keyword, checking is done at assembly time. Compatibility between the list and execute forms of the RACROUTE REQUEST=DIRAUTH macro is validated at execution time if you specify the `CHECK` subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the `RELEASE` keyword on the execute form of the macro. Otherwise, when `CHECK` processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.

The default is `RELEASE=1.6`.

- `MF=(E, ctrl addr)`

specifies the execute form of the RACROUTE REQUEST=DIRAUTH macro instruction.

**RACROUTE REQUEST=DIRAUTH (modify form)**

The modify form of the RACROUTE REQUEST=DIRAUTH macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=DIRAUTH macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

```
  name
  b
  RACROUTE
  b
  REQUEST=DIRAUTH
  \RTOKEN=message token addr
  \RESCSECLABEL=seclabel addr
  \ACCESS=READ
  \ACCESS=READWRITE
  \ACCESS=WRITE
  \ACEE=ACEE addr
```

- `name`: Symbol. Begin `name` in column 1.
- `b`: One or more blanks must precede RACROUTE.
- `RACROUTE`: One or more blanks must follow RACROUTE.
REQUEST=DIRAUTH

,USERSECLABEL= seclabel addr
    seclabel addr: Rx-type address or register (2) – (12)

,ACEEALET=ALET addr
    ALET addr: Rx-type address or register (2) – (12)

,CLASS=class name addr
    class name addr: Rx-type address or register (2) – (12)

,LOG=ASIS
    Default=ASIS

,LOG=NOFAIL

,LOG=NONE

,LOGSTR=logstr addr
    logstr addr: Rx-type address or register (2) – (12)

,TYPE=MAC
    Default=MAC

,TYPE=EQUALMAC

,TYPE=RVRSMAC

,MF=(M,ctrl addr)
    ctrl addr: Rx-type address or register (1) or (2) – (12)

The parameters are explained under the standard form of the RACROUTE REQUEST=DIRAUTH macro with the following exception:

,MF=(M,ctrl addr)
    specifies the modify form of the RACROUTE REQUEST=DIRAUTH macro instruction.

RACROUTE REQUEST=EXTRACT: Replace or retrieve fields

The RACROUTE REQUEST=EXTRACT macro retrieves or replaces certain specified fields from a RACF profile, encodes certain clear-text (readable) data, or creates an ENVR object representing the security environment from an existing ACEE.

The RACROUTE REQUEST=EXTRACT macro supports an SRB-compatible branch entry when you specify BRANCH=YES and TYPE=ENCRYPT, BRANCH=YES and TYPE=EXTRACT, or TYPE=ENVRXTR. In addition, cross memory mode is supported for TYPE=ENVRXTR, and for TYPE=EXTRACT with BRANCH=YES when running in system key and supervisor state.

Note: The area returned by a RACROUTE REQUEST=EXTRACT or EXTRACTN request is located below 16MB.

Caller restrictions:
1. This macro is available only to authorized callers (APF-authority, system key 0–7, or supervisor state).
2. Callers in cross-memory mode must be in primary ASC mode when they issue the RACROUTE request.
3. Cross-memory callers using TYPE=EXTRACT with BRANCH=YES or TYPE=ENVRXTR must be in system key and supervisor state.
4. Callers cannot hold any locks when issuing RACROUTE REQUEST=EXTRACT.

General restrictions:
1. When activated, automatic direction of application updates propagates RACROUTE REQUEST=EXTRACT,TYPE=REPLACE requests on to selected
remote nodes. Only RACROUTE REQUEST=EXTRACT;TYPE=REPLACE requests with return code 0 are propagated. Also, if your installation is set up such that password changes are propagated to other (target) nodes, you should be aware that attempts to change the password of revoked users on target nodes will fail. Thus if you are using automatic password direction or password synchronization, you should make sure that all target user IDs are resumed before changing the password.

2. Encoding, replacement, and extraction functions are mutually exclusive.

3. Logging of RACROUTE REQUEST=EXTRACT invocations is not done except indirectly. Logging can occur during field access checking if the RACROUTE REQUEST=AUTH request exit requests it. See “SMF Records” in z/OS Security Server RACF Macros and Interfaces for information about RACROUTE logging of Type 80 records.

4. TYPE=EXTRACTN is only supported by branch-entered EXTRACT when looking for a discrete profile in a general resource class that has been RACLISTed to a dataspace.

5. Callers of branch-entered EXTRACT running in cross memory mode are supported for TYPE=EXTRACT for general resource classes, and for TYPE=ENVRXTR. Other requests in cross memory mode will cause unpredictable results.

Programming interface information:

1. Use caution when using these interfaces as general programming interfaces for product code because they might not be supported by security products other than RACF.

2. For REQUEST=EXTRACT, the following functions are the only recommended programming interfaces:
   a. Retrieving or updating fields in any other product segment (including WORKATTR) in the user, group, and resource profiles.
   b. Retrieving the following installation-reserved fields:
      • USERDATA
      • USRCNT
      • USRDATA
      • USRFLG
      • USRN
   c. Retrieving the current or a specified user's default group or password.
   d. Retrieving the member list from a SECLABEL profile.

3. The following functions of RACROUTE REQUEST=EXTRACT are programming interfaces, but are not recommended.
   • Retrieving or updating fields, other than the APPLDATA field, in the BASE segment of a user, resource, or group profile. The APPLDATA field can be retrieved and updated.

To see the names of database fields that you can retrieve and update, refer to the database template listings in Appendix B, “RACF database templates,” on page 347; these listings show the valid segment and field names, and the basic information content of the fields. It shows also what is and what is not part of the product interface.

Guidelines:

1. To read information from the RACF database, use any of the following:
   • Output from database unload utility (IRRDBU00).
REQUEST=EXTRACT

- A relational database created from the IRRDBU00 output.
- The profile extract functions of the SAF R_admin callable service. (Data set profiles are not supported.)

When these are not feasible, consider using RACROUTE REQUEST=EXTRACT,TYPE=EXTRACT.

2. To update information in the RACF database, use the RACF command processors. You can execute the RACF commands in a batch environment using one of the following services:
   - TSO/E IKJEFTSR service
   - SAF R_admin callable service

When these are not feasible, consider using RACROUTE REQUEST=VERIFY (to change a user's password) or RACROUTE REQUEST=EXTRACT,TYPE=REPLACE.

For updates that RACROUTE REQUEST=EXTRACT,TYPE=REPLACE cannot process (such as database fields that exist in repeat groups, or encrypted fields where there is no plain-text data), you may need to use the ICHEINTY macro. However, use the command processors whenever possible.

Restriction for multilevel secure environments: When SETROPTS SECLBYSYSTEM is in effect, security label names extracted using BRANCH=YES with TYPE=EXTRACTN will contain a lowercase character in place of the first character of inactive security label names. When the first character is a national character, such as $ (X'5B'), # (X'7B'), or @ (X'7C'), the 40 bit will be turned off.

RACROUTE REQUEST=EXTRACT (standard form)

The standard form of the RACROUTE REQUEST=EXTRACT macro is written as follows. For a description of additional keywords that you can code and additional parameters that are required on the RACROUTE request, but that are not specific to this request type, see "RACROUTE (standard form)" on page 13.

Note:

RACROUTE REQUEST=EXTRACT requires an ACEE. For most applications, the system will have created an ACEE to represent the active user. However, for special cases where no ACEE exists, the invoker must create one before invoking RACROUTE REQUEST=EXTRACT.

The most common way to create an ACEE is to issue a RACROUTE REQUEST=VERIFY, specifying ENVIR=CREATE. After all RACROUTE invocations are complete, the invoker should issue RACROUTE REQUEST=VERIFY with ENVIR=DELETE specified, to delete the ACEE previously created.

Application programs must be structured so that a task requesting RACF services does not do so while other I/O initiated by the task is outstanding. If such I/O is required, the task should either wait for the other I/O to complete before requesting RACF services, or the other I/O should be initiated under a separate task. This is necessary to assure proper processing in recovery situations.

name: Symbol. Begin name in column 1.

b: One or more blanks must precede RACROUTE.
REQUEST=EXTRACT

RACROUTE

One or more blanks must follow RACROUTE.

REQUEST=EXTRACT

,TYPE=EXTRACT
,TYPE=EXTRACTN
,TYPE=REPLACE
,TYPE=ENCRYPT
,TYPE=ENVRXTR

,ACEE=acee addr

ACEE: A-type address or register (2) – (12)

,ENTITY=profile name addr

ENTITY: A-type address or register (2) – (12)

,ENTITYX=extended profile name addr

ENTITYX: A-type address or register (2) – (12)

,FLDACC=YES

Default: FLDACC=NO

,FLDACC=NO

,GENERIC=ASIS

Default: GENERIC=ASIS

,GENERIC=YES

,VF=S

If TYPE=EXTRACT or EXTRACTN is specified

,BRANCH=YES

Default: BRANCH=NO

,BRANCH=NO

,CCLASS='class name'

CLASS: 1–8 character name

,CCLASS=class name addr

CLASS: A-type address or register (2) – (12)

Default: CLASS='USER'

,DATFMT=YYYYDDDF

Default: DAtFMT=YYDDDF

,DATFMT=YYDDDF

,DERIVE=YES

See explanation of keyword.

Default: Normal processing

,FIELDS=field addr

FIELD: A-type address or register (2) – (12)

,MATCHGN=YES

Default: MATCHGN=NO

,MATCHGN=NO

,SEGMENT=segment name

SEGMENT: 1–8 character name

,SEGMENT=segment name addr

SEGMENT: A-type address or register (2) – (12)

,SUBPOOL=subpool number

SUBPOOL: Decimal digit 0–255

Default: See explanation of SUBPOOL keyword on page 114
REQUEST=EXTRACT

If TYPE=REPLACE is specified:

,CLASS='class name'
  class name: 1–8 character name
,CLASS=class name addr
  class name addr: A-type address or register (2) – (12)
  Default: CLASS='USER'

,DATEFMT=YYYYDDDF
  Default: DATEFMT=YYDDDF

,DATEFMT=YYDDDF

,FIELDS=field addr
  field addr: A-type address or register (2) – (12)

,SEGDATA=segment data addr
  segment data addr: A-type address or register (2) – (12)

,SEGMENT='segment name'
  segment name: 1–8 character name
,SEGMENT=segment segment name addr
  segment name addr: A-type address or register (2) – (12)

If TYPE=ENCRYPT is specified:

,BRANCH=YES
,BRANCH=NO
  Default: BRANCH=NO

,ENCRYPT=(data addr,DES)
  data addr: A-type address or register (2) – (12)

,ENCRYPT=(data addr,HASH)

,ENCRYPT=(data addr,INST)

,ENCRYPT=(data addr,STDDES)

Note: If TYPE=ENCRYPT is specified, the only other allowable parameters are ENTITY, ENTITYX, RELEASE, ENCRYPT, and BRANCH, with ENCRYPT being required.

If TYPE=ENVRXTR is specified:

,ENVROUT=envr data addr
  envr data addr: A-type address or register (2) – (12)

Note: When TYPE=ENVRXTR is specified, only the keywords ACEE and ENVROUT are recognized, with ENVROUT being required.

The parameters are explained as follows:

,ACEE=acee addr
  specifies an alternate ACEE for RACF to use rather than the current ACEE. For example, for the USER class or for CLASS= not specified, if the ENTITY or ENTITYX parameter has not been specified, or ENTITYX has been specified with zero for the buffer length and zero for the actual entity name length, RACF refers to the ACEE during extract processing of user data.

If you want to use the ACEE parameter, you must specify RELEASE=1.8 or later.

For TYPE=ENVRXTR, if the caller is in cross-memory mode, the specified ACEE must reside in the HOME address space.

For TYPE=EXTRACT,BRANCH=YES, if the caller is in cross-memory mode, the specified ACEE must reside in the PRIMARY address space.
REQUEST=EXTRACT

, BRANCH=YES
, BRANCH=NO

specifies whether you want RACF to use a branch entry.

The following applies to TYPE=EXTRACT with BRANCH=YES:

The RACROUTE REQUEST=EXTRACT macro supports an SRB-compatible branch entry when you specify BRANCH=YES and TYPE=ENCRIPT or BRANCH=YES and TYPE=EXTRACT with no change in function. with TYPE=EXTRACTN.

Cross memory mode is supported to obtain general resource profiles.

- General resource profiles that can be brought into storage are candidates for branch entry EXTRACT.
  - You can use the SETROPTS RACLIST command or RACROUTE REQUEST=LIST, GLOBAL=YES command to create a global listing of profiles in a data space. You can use this list only in the address space it was issued from.
  - You can also use RACROUTE REQUEST=LIST, GLOBAL=NO to create a listing of profiles in the user's address space, but this does not create a global listing of profiles.

- User data that is defaulted from the ACEE is a candidate for branch entry EXTRACT. This occurs when the USER class is specified or CLASS= is not specified, no ENTITY or ENTITYX is specified or ENTITYX is specified with zero for buffer length and zero for the actual entity name length, and no SEGMENT or FIELDS information is specified. The user's ID and default connect group are extracted from the current ACEE.

If the user's primary and secondary languages are available, they are also extracted from the current ACEE, along with a code (U) indicating that the reported languages are defined in the user's profile. If the user's primary and secondary languages are not available in the user's profile, the installation default primary and secondary languages that are set by SETROPTS are returned, along with a code (S) indicating that the reported languages are the installation default.

Additionally, if the user's work attributes (WORKATTR) information is available, it is also extracted from the ACEE. For the format of the WORKATTR information that is returned from the ACEE, see “RXTW: RACROUTE REQUEST=EXTRACT Result Area Mapping” in Server RACF Data Areas.

- RACF can extract the following fields of the general-resource profile:

  NOT programming interface information

  CATEGORY, IPLOOK, MEMCNT, MEMLST, and NUMCTGY. **Exception:** The MEMCNT and MEMLST fields of the SECLABEL profile are programming interfaces.

  End of NOT programming interface information

ACL2, ACL2ACC, ACL2CNT, ACL2NAME, ACL2RSVD, ACL2UID, ACLCNT, APPLDATA, AUDIT, CONVSEC, CSFAUSE, CSFCPW, CSFEXP, CSFSLBS, CFSCLCT, CFSKLB, CSFSLCT, DIDCT, DIDLIST1, GAUDIT, INSTDATA, KEYDATE, KEYINVT, LEVEL, LOGDAYS, LOGTIME, LOGZONE, NOTIFY, OWNER, SECLABEL, SECELEVEL, SESSKEY, SLSFLAGS, UACC, USERACS, USERID, and WARNING.

- RACF searches RACTION entries in the following order:
REQUEST=EXTRACT

- Those off the ACEE (if ACEE is specified),
- Those off the TCB ACEE in the PRIMARY address space,
- Those off the ASXB ACEE in the PRIMARY address space.

If the profile is not found off any ACEE, RACF searches globally RACLISTed profiles.

To specify the BRANCH keyword, you must specify Release=1.9 or later.

CLASS='class name'
CLASS=class name addr

specifies the class the entity is in. The class name can be USER, GROUP, CONNECT, DATASET, or any general-resource class defined in the class descriptor table.

If you specify CLASS, you must specify RELEASE=1.8 or later.

DATEFMT=YYYYDDDF
DATEFMT=YYDDDF

specifies the format of the date that you want to extract or replace. If you specify DATEFMT=YYYYDDDF and TYPE=EXTRACT or EXTRACTN, RACF retrieves date fields in the format ccyydddF where cc=19 or cc=20, unless the date being retrieved is in a uninitialized state in the RACF database, in which caseX'0000000F' or X'FFFFFFFF' is returned. If TYPE=REPLACE is specified, RACF accepts dates in the format ccyydddF where cc=19 or cc=20. When accepting a date as input to put into the database, RACF validates that cc=19 or cc=20, and that for cc=19, 70 < yy <= 99

and for cc=20, 00 <= yy <= 70.

If you specify DATEFMT=YYDDDF, RACF retrieves and accepts dates in the normal three byte format.

To specify the DATEFMT keyword, you must specify Release=1.9.2 or a later release number.

DERIVE=YES

specifies that the desired field be obtained from the DFP segment of the appropriate profile. To specify DERIVE, you must also specify RELEASE=1.8.1 or later.

DERIVE requests are limited to the DFP segment of the DATASET and USER profiles. The following explains the DERIVE processing for both a DATASET and a USER request:

- DATASET
  Specifying the DERIVE=YES keyword with CLASS=DATASET and FIELDS=RESOWNER causes RACF to perform additional processing other than simply extracting the data set resource owner from the data set profile. DFP uses this retrieved information for authority checking when allocating a new data set.
  To process the request, RACF first attempts to extract the RESOWNER field from the DATASET profile specified by the ENTITY or ENTITYX keyword. If the profile exists and the RESOWNER field contains data, RACF checks to see whether that data is the user ID of a user or group currently defined to RACF. If so, RACF returns that user ID along with a reason code that indicates whether the user ID is that of a user or a group.
If RACF does not find a profile that matches the DATASET name specified by the ENTITY or ENTITYX keyword, RACF attempts to locate the generic DATASET profile that protects that DATASET name.

If it finds the generic profile, and the RESOWNER field contains data, RACF checks to see whether that data is the user ID of a user or a group currently defined to RACF. If so, RACF returns that user ID along with a reason code that indicates whether the user ID is that of a user or a group.

If RACF does not find a generic profile, or the retrieved data is neither a user or group, RACF returns the high-level qualifier from the name specified on the ENTITY or ENTITYX keyword along with a reason code that indicates whether that high-level qualifier matches a defined user or group, or neither.

You specify a DERIVE request for RESOWNER as follows:

```
RACROUTE REQUEST=EXTRACT, TYPE=EXTRACT, X
ENTITY=DSNAME, X
VOLSER=MYDASD, X
CLASS='DATASET', X
FIELDS=RESFLDS, X
SEGMENT='DFP', X
DERIVE=YES, X
WORKA=RACWK, X
RELEASE=1.8.1
```

Note: You must specify all the keywords in the example for the DERIVE request to work.

- **USER**

The purpose of specifying the DERIVE=YES keyword with CLASS=USER is to obtain the desired DFP field information (STORCLAS, MGMTCLAS, DATACLAS or DATAAPPL) from the profile of the user. If the user's profile does not contain the desired DFP fields, RACF goes to the user's default group and attempts to obtain the information for the remaining fields from the GROUP profile (the remaining fields being those that do not contain information in the USER profile).

You specify a DERIVE request for information from a USER profile as follows:

```
RACROUTE REQUEST=EXTRACT, TYPE=EXTRACT, X
ENTITY=USER01, X
CLASS='USER', X
FIELDS=STRFLDS, X
SEGMENT='DFP', X
DERIVE=YES, X
WORKA=RACWK, X
RELEASE=1.8.1
```

Note: You must specify all the keywords in the example for the DERIVE request to work.
REQUEST=EXTRACT

RACF processes the DERIVE keyword if it is specified with the DATASET or USER class. In addition, for DERIVE processing to occur, SEGMENT=DFP and RELEASE=1.8.1 or later must also be specified.

\,\text{ENCRYPT=}(\text{data addr}, \text{DES})
\,\text{ENCRYPT=}(\text{data addr}, \text{HASH})
\,\text{ENCRYPT=}(\text{data addr}, \text{INST})
\,\text{ENCRYPT=}(\text{data addr}, \text{STDDES})

specifies the user-authentication key and authentication method.

\textbf{Note:} If the password was shipped from another system, the encryption method must be the same on all systems utilizing the password. For example, the RACF password authentication exits, ICHDEX01 and ICHDEX11, must be identical on all systems.

Specifying zero for the 1-byte length associated with the user-authentication key has the same effect as not specifying the keyword. Upon return to the caller, the first subparameter contains the address of an area that contains a 1-byte length followed by the product of the authentication process. Neither the address itself nor the length is changed. Also, data is one-way transposed; that is, no facility is provided to recover the data in readable form.

- \,\text{ENCRYPT=}(\text{data addr}, \text{DES})

  Specifies the user-authentication key and the National Bureau of Standards Data Encryption Standard (DES) encryption method. The address points to a 1-byte length followed by from 1 to 255 bytes of text to be used as the key for encryption. The second subparameter specifies the RACFDES algorithm (RACF's variation of DES). When the DES algorithm is used, RACF uses the variable-length user-authentication key to encrypt eight bytes of clear-text data pointed to by the ENTITY or ENTITYX parameter, or by the user ID from the current ACEE (if no ENTITY or ENTITYX is specified or if ENTITYX is specified with zero for the buffer length and zero for the actual entity-name length).

  RACF uses the first eight bytes of clear-text data pointed to by the ENTITY or ENTITYX parameter, or the user ID specified in the ACEE. All other text is ignored.

- \,\text{ENCRYPT=}(\text{data addr}, \text{HASH})

  Specifies the user-authentication key and the RACF hashing algorithm. The address points to a 1-byte length followed by from 1 to 255 bytes of text to be used as the user-authentication key. The second subparameter specifies the RACF hashing algorithm. When this hashing algorithm is used, the user-authentication key is masked instead of encrypted.

- \,\text{ENCRYPT=}(\text{data addr}, \text{INST})

  Specifies the user-authentication key and the INST authentication method. The address points to a 1-byte length followed by from 1 to 255 bytes of text to be used as the key for authentication. The second subparameter specifies whatever scheme the installation is using (INST value). When the INST algorithm is used, RACF passes to the installation-defined algorithm the variable-length user-authentication key and the eight bytes of clear-text data pointed to by the ENTITY or ENTITYX parameter, or by the user ID from the current ACEE (if no ENTITY or ENTITYX is specified or if ENTITYX is specified with zero for the buffer length and zero for the actual entity-name length).

  If there is no installation-defined authentication method present, RACF uses the DES encryption method. RACF uses the first eight bytes of clear-text
REQUEST=EXTRACT

data pointed to by the ENTITY or ENTITYX parameter, or the user ID specified in the ACEE. All other text is ignored.

• ,ENCRYPT=(data addr,STDDES)
  Specifies the user-authentication key and the STDDES authentication method. The address points to a 1-byte length followed by eight bytes of text to be used as the key for authentication. The second subparameter specifies the NBS DES algorithm. When the STDDES algorithm is used, RACF uses the 8-byte user authentication key to encrypt eight bytes of clear-text data pointed to by the ENTITY or ENTITYX parameter, or the user ID from the current ACEE if no ENTITY or ENTITYX is specified or if ENTITYX is specified with zero for the buffer length and zero for the actual entity-name length.

  The authentication key must be eight bytes in length. Any other length for the key results in a parameter-list error. RACF uses the first eight bytes of clear-text data pointed to by the ENTITY or ENTITYX parameter, or the user ID specified in the ACEE. All other text is ignored.

  ,ENTITY=profile name addr
  ,ENTITYX=extended profile name addr
  specifies the address:

  • ,ENTITY=profile name addr
    For Release 1.7 or earlier (limited to USER), specifies the address of an area eight bytes long that contains the resource name (user ID for CLASS=USER) for which profile data is to be extracted, or the user ID to be used when encoding. The name must be left-justified in the field and padded with blanks. If this parameter is not specified for TYPE=EXTRACT, a default value of zero indicates that RACF should use the user ID provided in the ACEE operand. For TYPE=REPLACE, the parameter must be specified. If CLASS=USER is coded, information from the ACEE control block is returned in the result area.

    For Release 1.8 and later, specifies the address of a resource name for which profile data is to be extracted or replaced for TYPE=EXTRACT, or REPLACE, or the clear-text data to be processed for TYPE=ENCRYPT. The area is 8 bytes long for USER and GROUP, 17 bytes long for CLASS=CONNECT, and 44 bytes long for DATASET. The lengths of all other profile names are determined by the class descriptor table. The name must be left-justified in the field and padded with blanks. If this parameter is not specified for TYPE=EXTRACT, a default value of zero indicates that RACF should use the user ID provided in the ACEE operand. For TYPE=REPLACE, the parameter must be specified. If CLASS=USER is coded, information from the ACEE control block is returned in the result area.

  Note: When you specify MATCHGN=YES, if your RACF installation is not using the restructured-database format and the length of an entity name for a general-resource class is longer than 39 characters, RACF uses generic profiles to match the name.

  • ,ENTITYX=extended profile name addr specifies the address of a structure that consists of two 2-byte length fields, followed by the entity name.
    – The first 2-byte field specifies a buffer length that can be from 0 to 255 bytes. This length field refers to the length of the buffer that contains the entity name; it does not include the length of either length field.
REQUEST=EXTRACT

- The second 2-byte field specifies the actual length of the entity name. This length field includes the length of the actual name without any trailing blanks; it does not include the length of either length field.

These two length fields can be used in several different ways:
- If you know the length of the entity name, you can specify 0 in the first field and the length of the entity name in the second field. When you specify the second field, note that each byte counts. This means the entity name that you specify must match exactly the entity name on the RACF database.
- If you choose to place the entity name in a buffer area, specify the length of the buffer in the first field. For the second field, do one of the following:
  - If you know the length of the entity name, specify the length in the second field. The length of the first field can be from 0–255 bytes, but must be equal to or greater than the length of the second field.
  - If you do not know the length of the entity name, specify 0 in the second field, and have RACF determine the number of characters in the entity name.

If this parameter is not specified or is specified in one of the following formats, a default value of zero indicates that RACF should use the user ID from the current ACEE. These are the only two situations in which specifying zero for the buffer length and zero for the actual entity-name length does not result in a parameter-list error.
- Zero is specified for the buffer length and zero is specified for the actual entity-name length for TYPE=ENCRIPT.
- Zero is specified for the buffer length and zero is specified for the actual entity-name length for TYPE=EXTRACT with USER being specified for the class or CLASS= being unspecified.

Note:
1. When you specify MATCHGN=YES, if your RACF installation is not using the restructured-database format and the length of an entity name for a general-resource class is longer than 39 characters, RACF uses generic profiles to match the name.
2. Programs using RACROUTE REQUEST=EXTRACT and ENTITY=, and either TYPE=EXTRACTN or TYPE=EXTRACT (with MATCHGN=YES) with an ENTITY buffer shorter than the class's MAXLENX value, might experience unpredictable results if the programs retrieve a profile name longer than the class's MAXLNTH value that does not fit within the buffer. Errors might include immediate 0C4 abends due to overlays of program storage. (MAXLNTH and MAXLENX are operands on the ICHERCDE macro. See z/OS Security Server RACF Macros and Interfaces for further information about the ICHERCDE macro.)

Guideline: Use ENTITYX rather than ENTITY. With ENTITY, the entity name you pass to RACF must be in a buffer, the size of which is determined by the length in the class descriptor table. If the maximum length of a class descriptor entity increases in the future, you must modify your program to use a larger buffer. By using ENTITYX, you avoid this possible problem because you removed the class descriptor table dependency from your program.

Restriction: When using BRANCH=YES with TYPE=EXTRACTN, the entity value should contain blanks, or a name returned by a previous extract. Passing an entity
name that does not exist in the RACLISTed profiles will result in an error rather than finding the next profile name alphabetically after the specified entity name.

\texttt{,ENVROUT=envr
data
daddr}

specifies the data structure that is to hold the information that is used to describe a security environment.

This information can be used later on a service with the ENVSHIFT keyword such as RACROUTE REQUEST=VERIFY to re-create the security environment without causing I/O to the RACF Database. The address points to a data structure defined in Table 5. The data structure describes the storage location for the ENVR object. The key of the ENVR data structure is a single-byte value that represents the associated ENVR object storage area. The low-order nibble of this value is the storage key. A key value of X'07' returns an ENVR object in key 07 storage.

The ENVR object extracted from the ACEE includes the X500 name, USP, ICTX, and IDID if they exist. Although the format of the data structure that ENVR points to is known to the RACROUTE callers, the content of the object itself is defined by the external security product.

This keyword is only recognized when TYPE=ENVRXTR is specified. To use this keyword, RELEASE=2.6 or later must also be specified.

Figure 3 and Table 5 represent the ENVR data structure for use with the ENVR ROUT keyword. The data structure must start on a fullword boundary.

<table>
<thead>
<tr>
<th>Description</th>
<th>Length (bytes)</th>
<th>ENVROUT usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENVR object length</td>
<td>4</td>
<td>Output</td>
</tr>
<tr>
<td>ENVR object storage area length</td>
<td>4</td>
<td>Input/output</td>
</tr>
<tr>
<td>ENVR object storage area address</td>
<td>4</td>
<td>Input/output</td>
</tr>
<tr>
<td>ENVR object storage area subpool</td>
<td>1</td>
<td>Input</td>
</tr>
<tr>
<td>ENVR object storage area key</td>
<td>1</td>
<td>Input</td>
</tr>
</tbody>
</table>

The ENVR object storage area can be supplied by the caller or obtained by RACF. If supplied by the caller, it must be on a doubleword boundary and be associated with the job step task. If RACF obtains the storage area, it is on a doubleword boundary and is associated with the Job Step task. The storage is
allocated based on the mode of the caller (LOC=ANY for 31-bit callers and LOC=BETWEEN for 24-bit callers). The following table shows how the field values affect ENVROUT processing.

Table 6. ENVROUT storage area processing

<table>
<thead>
<tr>
<th>ENVR object storage area length</th>
<th>ENVR object storage area address</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero</td>
<td>Any value</td>
<td>RACF obtains minimum storage size needed to contain the ENVR Object. Storage size is returned in the ENVR Object storage area length. Storage address is returned in the ENVR Object storage area address.</td>
</tr>
<tr>
<td>Nonzero</td>
<td>Zero</td>
<td>RACF obtains storage size specified or the minimum storage size needed to contain the ENVR Object. Storage size is returned in the ENVR Object storage area length. Storage address is returned in the ENVR Object storage area address.</td>
</tr>
<tr>
<td>Nonzero</td>
<td>Nonzero</td>
<td>RACF attempts to use the storage area provided. If the area is too small to contain the ENVR object, RACF frees the storage area provided and obtains the minimum storage size needed to contain the ENVR object. Storage size is returned in the ENVR object storage area length. Storage address is returned in the ENVR object storage area address.</td>
</tr>
</tbody>
</table>

Storage is obtained and freed in the subpool and key specified in the ENVR data structure.

Since the ENVR object length is returned to the caller, the ENVR object can be moved from one storage area to another. This value is supplied as an output to the caller. RACF does not attempt to use this value in either ENVROUTE or ENVRIN processing.

The caller is responsible for freeing the ENVR object storage area when it is no longer needed. The length, address, subpool, and key to be used when doing the FREEMAIN are contained in the ENVR data structure.

The ENVR object returned by ENVROUTE is a relocatable object that can be copied from one storage location to another. The returned ENVR object, or a copy of the returned ENVR object, can be specified as input to the RACROUTE interface using the ENVRIN keyword, or to the initACEE callable service using the ENVR_in parameter.

The ENVR object can be passed to other systems, but this should be done with great care. The ENVR object should not be saved for a long period of time before being used as ENVRIN, and it should not be passed to systems that have different security information. The other systems should share the RACF database and have compatible RACF installation exits and class descriptor tables.

,FIELDS=field addr

specifies the address of a variable-length list. The first field is a 4-byte field that contains the number of profile field names in the list that follows.
REQUEST=EXTRACT

Specifying zero for this 4-byte field has the same effect as not specifying the keyword.

Each profile field name is eight bytes long, left-justified, and padded to the right with blanks. The allowable field names for each type of profile are in the template listings in Appendix B, "RACF database templates," on page 347. For an illustration of how to specify the FIELDS keyword, see the TYPE=REPLACE example ("Example 1") on page 111.

For Release=1.8 or later, the following options exist:
- The count field can contain numbers 1–255.
- The field names can be any of the field names in the template listings, unless BRANCH=YES is specified. For more information, see the explanation of the BRANCH parameter.

If you specify TYPE=EXTRACT or EXTRACTN, RACF retrieves the contents of the named fields from the RACF profile indicated by the CLASS= and ENTITY= or ENTITYX= parameters, and returns the contents in the result area. (See the EXTRACT keyword for an explanation of the result area.)

Beginning with Release 1.8, you can specify TYPE=REPLACE. RACF replaces or creates the indicated fields in the profile specified on the CLASS and ENTITY or ENTITYX keywords with the data pointed to by the SEGDATA keyword.

Note:
1. Do not replace a repeat group-count field. Doing so causes unpredictable results.
2. You cannot replace an entire repeat group, a single occurrence of a repeat group, or a single existing field in a repeat group. If you attempt to do so, RACF adds the data to the existing repeat group or groups.
   The only things that you can do is retrieve all occurrences of specified fields within a repeat group, or add a new occurrence of a repeat group.
3. If you add occurrences of a repeat group, RACF places those additions at the beginning of the repeat group.

Example 1: This example of TYPE=REPLACE replaces fields in the BASE segment. It shows one way to code the macro and the declarations necessary to make the macro work.

```
RACROUTE REQUEST=EXTRACT,TYPE=REPLACE,
   CLASS='USER',
   ENTITY=USERID,
   FIELDS=FLDLIST,
   SEGDATA=SEGDLIST,
   SEGMENT=BASE,
   WORKA=RACWK
```

```
USERID DC CL8,'BILL'
FLDLIST DC A(3)
   DC CL8'AUTHOR'
   DC CL8'DFLTGRP'
   DC CL8'NAME'
SEGDLIST DC AL4(6),CL6'DJONES'
   DC AL4(8),CL8'SECURITY'
   DC AL4(11),CL11'BILL THOMAS'
BASE DC CL8'BASE'
RACWK DC CL512
```

When the replacement action takes place, the following occurs:
REQUEST=EXTRACT

- "DJONES" is placed in the AUTHOR field in the profile.
- "SECURITY" is placed in the DFLTGRP field in the profile.
- "BILL THOMAS" is placed in the 'NAME' field in the profile.

Example 2: In this example, RACROUTE REQUEST=EXTRACT retrieves the UACC from a fully qualified, generic data set profile. RACROUTE places the information in a work area in subpool 1.

```
RACROUTE REQUEST=EXTRACT,TYPE=EXTRACT,
   X
   VOLSER=VOLID, X
   CLASS='DATASET', X
   ENTITY=DSN, X
   FIELDS=FLDS, X
   GENERIC=YES, X
   SUBPOOL=1, X
   RELEASE=1.8, X
   SEGMENT='TSO', X
   WORKA=RACWK

DSN DC CL44'SYS1.LINKLIB'
FLDS DC A(1)
   DC CL8 'UACC'
RACWK DC CL512
```

,FLDACC=NO
,FLDACC=YES

specifies whether field-level access checking should be performed.

If you specify FLDACC=YES, the RACF database manager checks to see that the user running your program has the authority to extract or modify the fields specified in the RACROUTE REQUEST=EXTRACT macro.

Note:
1. FLDACC=YES is ignored if BRANCH=YES is specified.
2. For field-level access checking to occur, you must specify RELEASE =1.8 or later when you code the macro. In addition, before the program executes, the security administrator must activate and RACLIST the FIELD class using the SETROPTS command. If you code FLDACC=YES and the FIELD class is not active and RACLISTed, the request is failed with a return code 8, reason code 4.
3. In addition, the security administrator must issue the RDEFINE and PERMIT commands to designate those users who have the authority to access the fields designated in the RACROUTE REQUEST=EXTRACT macro.
4. If you specify FLDACC=NO or omit the parameter, the manager ignores field-level access checking.
5. If you specify FLDACC=YES for TYPE=REPLACE with segment name CSDATA and field name CSCDATA, the data associated with the CSKEY field is used instead of the template name for field level access checking. This allows field level access checking to use the FIELD profile USER.CSDATA.custom-field-name or GROUP.CSDATA.custom-field-name for authorization, similar to the authorization that occurs for the RACF commands.

,GENERIC=ASIS
,GENERIC=YES

specifies whether RACF is to treat the entity name as a generic profile name.

YES RACF considers the entity name a generic profile name, even if it does
REQUEST=EXTRACT

not contain any of the generic characters. If you specify GENERIC=YES, the resource name in the macro will match only a generic resource name in the RACF database. It will not match a discrete name.

Characters considered generic are:

- For data set class:
  - Asterisk (*)
  - Percent (%)
- For general resource class:
  - Asterisk (*)
  - Percent (%)
  - Ampersand (&)

**ASIS**

RACF considers the entity name a generic profile name if it contains:

- For data set class:
  - Asterisk (*)
  - Percent (%)
- For general resource class:
  - Asterisk (*)
  - Percent (%)
  - Ampersand (&).

**Note:** A profile in the RACFVARS class is not considered to be a generic profile even though it contains an ampersand sign.

If you specify GENERIC, you must specify RELEASE=1.8 or later.

**Note:** If generics have not been enabled for the class, EXTRACT and EXTRACTN do not find generic profiles.

\texttt{,MATCHGN=\texttt{YES},MATCHGN=\texttt{NO}}

specifies that you want to extract data from a profile that matches or covers the resource name specified on the ENTITY or ENTITYX keyword.

**YES**

RACF extracts data from the discrete profile, if one exists; if a discrete profile does not exist, RACF extracts data from the best-matching generic profile. If a best-matching generic profile is found, that profile name is returned to the caller in the ENTITY or ENTITYX location.

If ENTITYX is specified, the length of the best-matching profile name is also returned in the 2-byte, actual entity-name-length location. If the buffer length is less than the length of the best-matching profile, you get a return and reason code indicating that the profile was not found because the buffer length specified is too small.

**NO**

RACF extracts data from a profile (discrete or generic) that exactly matches the name specified on the ENTITY or ENTITYX keyword.

**Note:**

1. To specify the MATCHGN keyword, you must specify Release=1.9 or a later release number.
2. For MATCHGN=YES, the class must be active.
REQUEST=EXTRACT

,SEGDATA=segment data addr
  specifies the address of a list of data items to be placed in the respective fields
  named by the FIELDS= parameter. You use the SEGDATA parameter when you
  specify TYPE=REPLACE. If you specify SEGDATA, you must also specify
  CLASS, FIELDS, and RELEASE=1.8 or a later release number. The stored data
  is paired in the following format:
  • A 4-byte length field that contains the length of the data field that follows
  • A data field of variable length.

  Specifying zero for the length field causes the field being replaced to be
  removed from the segment. Each length field is followed immediately by a
  data field, until the end of the replacement data is reached. The count field,
  which is pointed to by the first field in the FIELDS parameter, contains the
  total number of length-data pairs.

,SEGMENT='segment name'

,SEGMENT=segment name addr
  specifies the RACF profile segment that RACF is to update or from which it is
  to extract data. If you allow the SEGMENT parameter to default, RACF
  assumes that you want to extract information from the base segment.

  Each segment name is eight bytes long, left justified, and padded to the right
  with blanks. SEGMENT is not preceded by a 4-byte length field.

  If you specify SEGMENT, you must also specify the CLASS and FIELDS
  keywords, and RELEASE=1.8 or a later release number.

,SUBPOOL=subpool number
  specifies the storage subpool from which the extract-function routine obtains
  an area needed for the extraction.

Note:
1. If this parameter is not specified, it defaults to 229.
   Take some care in selecting a subpool, as MVS makes certain assumptions
   about subpool usage and characteristics. In particular, using subpool 0 or
   250, or any subpool documented in z/OS MVS Programming: Assembler
   Services Guide as having a storage key of USER (for example, 227–231 and
   241) might give unpredictable results.

2. If a common-area subpool (for example 226–228, 231, 239, 241, 245, 247, or
   248) is used and not freed before the job terminates, then the job might
   show up in the exception reports of RMF (or other monitoring tools that
   support the tracking of common-area storage utilization) as owning
   common storage. Before your job terminates, it should issue a FREEMAIN
   to free this common storage.

,TYPE=ENCRYPT
,TYPE=ENVRXTR
,TYPE=EXTRACT
,TYPE=EXTRACTN
,TYPE=REPLACE
  specifies the type of function to be performed by the extract-function routine.

ENCRYPT
  allows RACF to provide an authentication token to be used in
  verifying a user's identity. The ENCRYPT keyword specifies the
  user-authentication key to be used for authentication, and the
  authentication method. The first eight bytes of the area pointed to by
  the ENTITY or ENTITYX operand are used as the clear-text data to be
REQUEST=EXTRACT

processed by the INST, DES, and STDDES authentication routines. The HASH method uses the user-authentication key as clear-text data and masks the data instead of encrypting it. If ENTITY or ENTITYX is not specified, or ENTITYX is specified with zero for the buffer length and zero for the actual entity-name length, the user ID from the current ACEE is used as the clear-text data. If TYPE=ENCRYPT is specified, no work area is returned.

ENVRXTR

extracts an ENVR object from the specified ACEE. The ENVR object is returned using the ENVROUT keyword. TYPE=ENVRXTR can be invoked in cross-memory mode to extract an ENVR object in the primary address space from an ACEE in the HOME address space.

The ENVR object returned by ENVROUT= is a relocatable object that can be copied from one storage location to another. The returned ENVR object, or a copy of the returned ENVR object, can be specified as input to the RACROUTE interface using the ENVRIN keyword, or to the initACEE callable service using the ENVR_in parameter.

The ENVR object can be passed to other systems, but this should be done with great care. The ENVR object should not be saved for a long period of time before being used as ENVRIN, and it should not be passed to systems that have different security information. The other systems should share the RACF database and have compatible RACF installation exits and class descriptor tables.

Note:
1. When TYPE=ENVRXTR is specified, only the keywords ACEE and ENVROUT are recognized, with ENVROUT being required.
2. RELEASE=2.6 or later must be also specified
3. This service is always invoked as a branch entered service, and the BRANCH= keyword is ignored.
4. The issuer must be APF authorized, system key (0–7), or in supervisor state unless in cross memory mode. In cross memory mode, the caller must be system key (0–7) and in supervisor state.
5. This service is always invoked with DECOUPL=YES. The setting of the DECOUPL= parameter is ignored.

EXTRACT

Extract information from any field in any profile, unless BRANCH=YES is specified.

The profile templates in Appendix B, “RACF database templates,” on page 347 define the type and name of each field in each profile. If you specify EXTRACT, RACF extracts information from the profile determined by the ENTITY or ENTITYX and CLASS keywords.

Specifically, RACF extracts (from the RACF database) the fields specified in the FIELDS keyword from the segment specified by the SEGMENT keyword.

Otherwise, you can obtain the default user-class information from the current user’s profile (the specified or default ACEE) if you do the following:

- Specify the USER class or do not specify the CLASS= keyword.
- Do not specify the SEGMENT and FIELDS keywords.
REQUEST=EXTRACT

- Do not specify the ENTITY or ENTITYX keyword, or specify ENTITYX with zero for the buffer length and zero for the actual entity-name length.

An installation that is processing selected or all users in the USER class by coding RACROUTE REQUEST=EXTRACT,TYPE=EXTRACTN (where the starting profile is a value less than X'C1') might find that this code returns one of the irrcerta, irrmulti, and irrsitec user IDs, which are the user IDs associated with digital certificates. These user IDs are not valid user IDs and do not represent real users. Because they do not contain a default group, they cannot be used for RACROUTE REQUEST=VERIFY processing, nor can they be used on a third-party RACROUTE REQUEST=AUTH. As a result, an installation might want to ignore these user IDs in its code.

When the default information is taken from the current user's profile (the specified or default ACEE), there is no I/O to the RACF database, and the user's ID and default connect group are extracted from the current ACEE. This also results in returning the language information as follows:

- If the user's primary and secondary languages are available, they are extracted from the current ACEE, along with a code indicating that the reported languages are defined in the user's profile.
- If the user's primary and secondary languages are not available in the user's profile, the installation default primary and secondary languages set by SETROPTS are returned, along with a code indicating that the reported languages are the installation default.

Additionally, if the user's work attributes (WORKATTR) information is available, it is also extracted from the ACEE. For the format of the WORKATTR information returned from the ACEE, see “RXTW: RACROUTE REQUEST=EXTRACT Result Area Mapping” in z/OS Security Server RACF Data Areas.

To use TYPE=EXTRACT to extract field information from a profile, you must specify RELEASE=1.8 or a later release number.

Note: If you specify TYPE=EXTRACT, do not specify ENCRYPT.

Upon return, register 1 contains the address of a result area that begins with a fullword containing the length and subpool number of the area. EXTWOFF is used as the offset from EXTWKEA to locate the start of the optional returned fields. See “RXTW: RACROUTE REQUEST=EXTRACT Result Area Mapping” in z/OS Security Server RACF Data Areas for the mapping of this area. It is your responsibility to issue a FREEMAIN to release the area after you are through using it. See the description of the SUBPOOL keyword.

There are certain conditions under which register 1 can contain a non-zero value that is not a pointer to a result area.

<table>
<thead>
<tr>
<th>R15 value</th>
<th>SAFPRRET</th>
<th>SAFPRREA</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>0</td>
<td>anything</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>X'64'</td>
<td>anything</td>
<td>anything</td>
</tr>
</tbody>
</table>
REQUEST=EXTRACT

Otherwise, when register 1 contains a non-zero value, it contains the address of the result area.

You need to initialize the parameter lists such that SAFPRRET and SAFPRREA start out as 0 in order to distinguish these cases.

In general, RACF returns field data in the order it was specified, with a 4-byte length field preceding each profile field. The following lists show what is returned for different types of extractions:

- For a single field, you get:
  - A 4-byte length field that contains the length of the field that follows, or
  - If the requested field is a variable-length field, there is no additional length byte.

| 4 bytes (length of data) | data |

- For a combination field (representing one or more fields), you receive:
  - A 4-byte length field that contains the combined length of all the fields that follow, or
  - A combination field made up of 4-byte length fields followed by their respective individual data fields.

<table>
<thead>
<tr>
<th>Total length of combination field</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 bytes (length of data1)</td>
</tr>
<tr>
<td>4 bytes (length of data2)</td>
</tr>
</tbody>
</table>

- For a single field within a repeat group, you receive:
  - A 4-byte length field that contains the combined length of all the fields that follow, or
  - A 4-byte length field that indicates the length of the specified field in the first occurrence of the repeat group. This is followed by a 4-byte length field that indicates the length of the specified field in the second occurrence of the repeat group. This order repeats until all the occurrences of the repeat group are accounted for.

<table>
<thead>
<tr>
<th>Total length of all the following fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 bytes (length of data1)</td>
</tr>
<tr>
<td>4 bytes (length of data1)</td>
</tr>
</tbody>
</table>

- For a combination field (representing one or more fields) within a repeat group, you receive:
  - A 4-byte length field that contains the combined length of all the fields that follow, or
  - A combination field consisting of a 4-byte length field indicating the length of the individual data field that follows it, followed by the next 4-byte length field indicating the length of the next individual data field. This order repeats until all the individual fields that make up the combination field are accounted for. The order begins again for the next occurrence of the repeat group.
REQUEST=EXTRACT

| Total length of all the occurrences of the combination field in the repeat group. |
|---------------------------------|---------------------|
| 4 bytes (length of data1)       | data1               |
| 4 bytes (length of data2)       | data2               |
| 4 bytes (length of data1)       | data1               |
| 4 bytes (length of data2)       | data2               |

- When you specify the name of a repeat-group count field, you retrieve the 4-byte length followed by the 4-byte repeat group count.

When a field to be extracted is empty, the following results:
- For fixed length fields, RACF returns the default as specified by the template definitions. The default for flag fields is X'00'. The default for fixed-length fields in the BASE segment of the profile is binary 1(s). The default for fixed length fields in other segments is binary zeros.
- For variable-length fields, RACF returns a length of zero and no data.

**EXTRACTN**

specifies that upon return, register 1 contains the address of a result area that begins with a fullword containing the area’s subpool number and length. To see the format of the result area, see the explanation of TYPE=EXTRACT, above and “RXTW: RACROUTE REQUEST=EXTRACT Result Area Mapping” in z/OS Security Server RACF Data Areas. At offset 6 in the result area, there is a flag. If the flag has a X'80', the name returned is generic.

If you specify EXTRACTN, the macro extracts information from the profile that follows the profile determined by the ENTITY or ENTITYX and CLASS keywords. From that next profile, RACF extracts the fields specified in the FIELDS keyword from the segment specified by the SEGMENT keyword. In addition, RACF returns the name of the profile from which it extracted the data.

**Note:**
1. If you specify TYPE=EXTRACTN, do not specify ENCRYPT=.
2. To retrieve all profiles within a class, the database must be processed twice, once to extract all discrete profiles and once again to extract all generic profiles (see “Example 3” on page 124). In exception, the DATASET class needs to be processed only once to extract all discrete and generic profiles. (See “Example 4” on page 127.)
3.

**REPLACE**

Use of the REPLACE option to update a profile requires a thorough knowledge of the interrelationships of fields within a profile, and of the potential relationships between profiles. For instance, if you use RACROUTE REQUEST=EXTRACT to update a password, you should also update the password change date. However, since you cannot update the password history, subsequent password changes (by PASSWORD or TSO LOGON, for example) could allow the old password to be used again.
 REQUEST=EXTRACT

If you specify TYPE=REPLACE, RACF takes the information in the fields specified in the FIELDS parameter and pointed to by SEGDATA, and places that information in the designated segment. (The segment is within the profile determined by the ENTITY or ENTITYX and CLASS keywords.) If you specify TYPE=REPLACE, you must specify FIELDS, SEGDATA=, and RELEASE=1.8 or later. If you want to replace a segment other than the base segment, you must specify the SEGMENT keyword with the segment you want. If you do not specify SEGMENT, the segment defaults to the base segment.

With Release 1.8 and later, if you want to create a TSO segment, you can do so by specifying the RACROUTE REQUEST=EXTRACT macro in the following way:

```
TYPE=REPLACE  SEGMENT=TSO
```

Note: If you specify TYPE=REPLACE, do not specify ENCRYPT=.

,,VOLSER=volser addr

specifies the volume serial number as follows:

- For non-VSAM DASD data sets and for tape data sets, this specifies the volume serial number of the volume on which the data set resides.
- For VSAM DASD data sets and tape data sets, this specifies the volume serial number of the catalog controlling the data set.

The field pointed to by the VOLSER address contains the volume serial number. If necessary, you must pad it to the right with blanks so it contains six characters.

If you specify VOLSER, you must specify RELEASE=1.8 or later.

VOLSER is valid with CLASS=DATASET.

,,MF=S

specifies the standard form of the RACROUTE REQUEST=EXTRACT macro instruction.

Return codes and reason codes

When you execute the macro, space for the RACF return code and reason code is reserved in the first two words of the RACROUTE parameter list. You can access them using the ICHSAFP mapping macro, by loading the ICHSAFP pointer with the label that you specified on the list form of the macro. When control is returned, register 15 contains the SAF return code.

Note:

All return and reason codes are shown in hexadecimal. Also, note that SAF return code is presented as SAF RC and RACF return code is presented as RACF RC in the following section.

<table>
<thead>
<tr>
<th>SAF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACROUTE REQUEST=EXTRACT has completed successfully.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>The extraction, replacement, or encoding completed successfully.</td>
</tr>
</tbody>
</table>
### REQUEST=EXTRACT

#### Reason Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Some of the values are derived from the USER profile, and some might be derived from the GROUP profile.</td>
</tr>
<tr>
<td>04</td>
<td>High-level qualifier returned as RESOWNER, which matched a valid USER.</td>
</tr>
<tr>
<td>08</td>
<td>DFP data returned from an EXTRACT request from USER profile was actually from the user's default connect group.</td>
</tr>
<tr>
<td>0C</td>
<td>High-level qualifier returned as RESOWNER, which matched a valid GROUP.</td>
</tr>
<tr>
<td>24</td>
<td>RESOWNER field matched a valid USER.</td>
</tr>
<tr>
<td>28</td>
<td>RESOWNER field matched a valid GROUP.</td>
</tr>
<tr>
<td>2C</td>
<td>At least one, but not all, of the fields requested failed to be retrieved because of field-level access checking.</td>
</tr>
<tr>
<td>04</td>
<td>The requested function could not be performed.</td>
</tr>
</tbody>
</table>

#### RACF RC

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>No security decision could be made.</td>
</tr>
<tr>
<td>04</td>
<td>An ESTAE environment could not be established, or if Register 0 contains a reason code of 1, neither EXTRACT, EXTRACTN, REPLACE, nor ENCRYPT was specified for TYPE=.</td>
</tr>
<tr>
<td>08</td>
<td>For TYPE=EXTRACT, TYPE=EXTRACTN, or TYPE=REPLACE, the profile could not be found, or one of the other errors shown by the reason code has occurred.</td>
</tr>
<tr>
<td>00</td>
<td>No profile found.</td>
</tr>
<tr>
<td>04</td>
<td>Field-level access checking failed. The FIELD class might not be active and RACLISTed.</td>
</tr>
<tr>
<td>08</td>
<td>Segment not found.</td>
</tr>
<tr>
<td>0C</td>
<td>Class not RACLISTed (branch EXTRACT).</td>
</tr>
</tbody>
</table>
REQUEST=EXTRACT

10  Class not active (branch EXTRACT).

14  For EXTRACT:

Neither the RESOWNER field nor the high-level qualifier matched a valid USER or GROUP.

For branch-entered EXTRACT only:

Field specified is not valid. Note that the field can be valid for a profile in the RACF database, but not for a RACLISTed profile.

18  For EXTRACT:

For MATCHGN=YES with ENTITYX= specified, the buffer length specified was too small to return the matching generic profile.

For branch-entered EXTRACT only:

For TYPE=EXTRACTN with ENTITYX= specified, the buffer length specified was too small to return the next profile name.

1C  The class was RACLISTed by RACROUTE REQUEST=LIST, GLOBAL=YES, but the data space has been deleted (branch EXTRACT).

0C  RACF is inactive.

10  The extract operation failed. SAFPRREA contains the RACF-manager return code that caused the operation to stop. The value in the SAFPRREA should be interpreted as follows:

Reason Code

Meaning

xxxxxyyy

xxxx is the RACF manager reason code and yyyy is the RACF manager return code. This return code is not used for the encrypt function.

For branch EXTRACT, the return and reason codes in SAFPRREA are internal RACF codes indicating an error in RACF, except for the following:

Reason Code

Meaning

00080018

The RACLIST data space cannot be accessed due to an ALESERV failure.

Note: For an explanation of the RACF manager return codes, refer to the part of z/OS Security Server RACF Macros and Interfaces that discusses the ICHEINTY macro or the section of z/OS Security Server RACF Messages and Codes that describes RACF manager return codes.

14  For TYPE=ENCRYPT or TYPE=EXTRACT of USER class data, ENTITY or ENTITYX was not specified and no ACEE exists or the ACEE was not for a defined user.
### REQUEST=EXTRACT

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>No ACEE exists.</td>
</tr>
<tr>
<td>04</td>
<td>ACEERACF bit is off.</td>
</tr>
<tr>
<td>08</td>
<td>The requested function failed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>A parameter-list error was encountered.</td>
</tr>
</tbody>
</table>

### Reason Code

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>04</td>
<td>For a TYPE=REPLACE request, FIELDS= was not specified.</td>
</tr>
<tr>
<td>08</td>
<td>Type specified is not valid.</td>
</tr>
<tr>
<td>0C</td>
<td>Number of fields is not valid.</td>
</tr>
<tr>
<td>10</td>
<td>Class name specified is not valid.</td>
</tr>
<tr>
<td>14</td>
<td>Version in parameter list is not valid.</td>
</tr>
<tr>
<td>18</td>
<td>Subpool specified is not valid.</td>
</tr>
<tr>
<td>1C</td>
<td>Parameter length not valid.</td>
</tr>
<tr>
<td>20</td>
<td>For TYPE=REPLACE request, SEGDATA= was not specified.</td>
</tr>
<tr>
<td>24</td>
<td>Entity name specified is not valid.</td>
</tr>
<tr>
<td>2C</td>
<td>For TYPE=ENCRYPT request, no user-authentication key was specified.</td>
</tr>
<tr>
<td>30</td>
<td>Encoding method is not valid.</td>
</tr>
<tr>
<td>34</td>
<td>ENTITY= or ENTITYX= was not specified with TYPE=REPLACE, TYPE=EXTRACTN, or TYPE=EXTRACT with class other than USER.</td>
</tr>
<tr>
<td>38</td>
<td>Multiple profiles and no volume specified.</td>
</tr>
<tr>
<td>3C</td>
<td>Profile found, but the wrong volume serial number was specified.</td>
</tr>
<tr>
<td>40</td>
<td>For a TYPE=EXTRACT request of USER class data, FIELDS= is not permitted with BRANCH=YES because the USER class cannot be RACLISTERd (branch EXTRACT).</td>
</tr>
<tr>
<td>44</td>
<td>For the ENTITYX format, both the entity-name length and the buffer length are zero.</td>
</tr>
<tr>
<td>48</td>
<td>Entity-name length with the ENTITY or ENTITYX keyword is not valid:</td>
</tr>
<tr>
<td></td>
<td>• The specified length is less than zero.</td>
</tr>
<tr>
<td></td>
<td>• The specified length is one of the following:</td>
</tr>
<tr>
<td></td>
<td>• Greater than 44 if CLASS=DATASET,</td>
</tr>
<tr>
<td></td>
<td>• Greater than 8 if CLASS=USER or GROUP,</td>
</tr>
<tr>
<td></td>
<td>• Greater than 17 if CLASS=CONNECT, or</td>
</tr>
<tr>
<td></td>
<td>• Greater than the maximum for the specified class as defined in the class descriptor table.</td>
</tr>
</tbody>
</table>
REQUEST=EXTRACT

- For a TYPE=ENCRYPT request, the specified length is not zero or eight.

4C Buffer length specified with ENTITYX keyword not valid:
- Less than zero
- Greater than 255
- Not zero but less than the entity name length

50 The entity name contains a blank.
- If the ENTITYX keyword is specified and the entity-name length is given, the name has a blank in the beginning, in the middle, or at the end.

54 For a TYPE=ENCRYPT request for the STDDDES authentication method, the specified data length is not 8.

58 For a TYPE=EXTRACT request of user-class data, the user specified with ENTITY= or ENTITYX= is not the same as the user specified in the ACEE (branch EXTRACT).

5C For a TYPE=EXTRACT request of user-class data that is defaulted from the ACEE, FIELDS= and SEGMENT= are not permitted because the user ID in the ACEE is not that of a RACF-defined user.

60 For TYPE=ENVRXTR, the ENVROUT keyword was not specified.

64 For TYPE=ENVRXTR, the ENVR object storage area address specified was not on a doubleword boundary.

68 For TYPE=ENVRXTR, the ENVR object could not be built because the ACEE was not valid.

6C GENERIC=YES was specified with TYPE=EXTRACTN and BRANCH=YES.

70 MATCHGN=YES was specified with TYPE=EXTRACTN and BRANCH=YES.

64 Indicates that the CHECK subparameter of the RELEASE keyword was specified on the execute form of the RACROUTE REQUEST=EXTRACT macro; however, the list form of the macro does not have the same RELEASE parameter. It also indicates that the TYPE parameters specified on the list and execute forms might not be the same TYPE. Macro processing terminates.

Example 1
The following is an example of a RACROUTE REQUEST=EXTRACT that looks for an APPCLU profile that has been RACLISTed off an ACEE, the one called VTAMACEE, to match the entity named in LULUPAIR. Specifying BRANCH=YES means that the function does not invoke an SVC. It is therefore SRB-compatible. If the function finds a profile to match the entity name, it returns fields specified in FLDLST.

Because MATCHGN=YES is specified, if a discrete profile that matches the entity name is not found, the function looks for a best-matching generic profile. If a best-matching generic profile is found, the fields specified in FLDLST are returned from that profile. The best-matching generic profile is returned in the entity location.
**REQUEST=EXTRACT**

RACROUTE REQUEST=EXTRACT, TYPE=EXTRACT,
BRANCH=YES,
CLASS='APPCLU',
ENTITY=LULUPAIR,
ACEE=VTAMACEE,
SEGMENT='SESSION',
FIELDS=FLDLST,
MATCHGN=YES,
RELEASE=1.9,
WORKA=RACWK

LULUPAIR DC CL(35)'NETA.LU1.LU2'
FLDLST DC A(3)
DC CL8'SESSKEY'
DC CL8'KEYDATE'
DC CL8'KEYINTVL'
RACWK DC CL512

**Example 2**
The following is an example of a RACROUTE REQUEST=EXTRACT that uses the
STDDES authentication method to process the data in RANDATA, using the data
in SESSNKEY as the authentication key. The function overlays the data in
SESSNKEY with the product of the authentication process. Specifying
BRANCH=YES means that the function is compatible with SRB mode and does not
issue any SVCs.

RACROUTE REQUEST=EXTRACT, TYPE=ENCRYPT,
BRANCH=YES,
ENTITY=RANDATA,
ENCRYPT=(SESSNKEY,STDDES),
RELEASE=1.9,
WORKA=RACWK

RANDATA DC CLB'RANDATA1'
SESSNKEY DC AL1(8),CL8'SESSNKEY'
RACWK DC CL512

**Example 3**
The following is an example of a RACROUTE REQUEST=EXTRACT with
EXTRACTN. It retrieves all profiles within any class (except the DATASET class).
The database must be processed twice, once to extract all discrete profiles and once
to extract all generic profiles.

EXTRTNRGR CSECT
* Entry Linkage
* STM 14,12,12(13) Push caller registers
BALR 12,0 Establish ...
USING *,12 ... addressability
GETMAIN R,LV=DYNLEN Get dynamic storage
LR 11,1 Move getmained address to R11
USING DYNAREA,11 Addressability to DSECT
ST 13,SAVEAREA+4 Save caller save area address
LA 15,SAVEAREA Get address of own save area
ST 15,8(13) Store in caller save area
LR 13,15 Get address of own save area
*
* Initialize variables in dynamic storage area
*
* MVC ENTXBLEN,H6 Set buffer length to 6
MVC ENTXNLEN,H0 Set entity length to 0
REQUEST=EXTRACT

MVC ENTXNAME,BLNKNAME    Set entity name to blanks
*  
* Copy static RACROUTE to dynamic GETMAINed areas
*  
MVC DYNRACR(RACRLEN),STATRACR
*  
* Loop to retrieve the OWNER field from all discrete profiles in the TAPEVOL class.
*  
DISLOOP EQU * Start of discrete loop
RACROUTE REQUEST=EXTRACT,TYPE=EXTRACTN,ENTITYX=ENTXBUFF,   *
   FIELDS=FIELDS, WORKA=WORKAREA, RELEASE=1.9, MF=(E,DYNRACR)  
LTR 15,15    Check return code
BNZ TRYGEN  Exit on nonzero return code to search for generic profiles
*  
* Do discrete TAPEVOL profile processing here.
*  
*  
XR 3,3 Zero out register 3
XR 2,2 Zero out register 2
USING EXTWKEA,1 Base the result area on register 1
*  
ICM 3,1,EXTWSP Move the result area subpool into register 3
*  
ICM 2,7,EXTWLN Move the result area length into register 2
*  
DROP 1 Drop basing on register 1
FREEMAIN R, LV=(2), A=(1), SP=(3) Free storage before processing next profile
*  
B DISLOOP Process next discrete profile
*  
TRYGEN EQU * Search for generic profiles
*  
MVC ENTXBLN,H6 Set buffer length to 6
MVC ENTXLEN,H0 Set entity length to 0
MVC ENTXNAME,BLNKNAME Set entity name to blanks
SLR 15,15 Clear return code
*  
* Modify request to set GENERIC to YES
*  
RACROUTE REQUEST=EXTRACT,TYPE=EXTRACTN,GENERIC=YES,   *
   RELEASE=1.9, MF=(M,DYNRACR)  
*  
* Loop to retrieve the OWNER field from all generic profiles in the TAPEVOL class.
*  
GENLOOP EQU * Start of generic loop
*  
RACROUTE REQUEST=EXTRACT,TYPE=EXTRACTN,ENTITYX=ENTXBUFF,   *
   FIELDS=FIELDS, WORKA=WORKAREA, RELEASE=1.9, MF=(E,DYNRACR)  
LTR 15,15    Check return code
BNZ DONE  Exit on nonzero return code
*  
* Do generic TAPEVOL profile processing here.
*  
*  
XR 3,3 Zero out register 3
XR 2,2 Zero out register 2
USING EXTWKEA,1 Base the result area on register 1
REQUEST=EXTRACT

ICM 3,1,EXTWSP Move the result area subpool into register 3
* ICM 2,7,EXTWLN Move the result area length into register 2
* DROP 1 Drop basing on register 1
FREEMAIN R,LV=(2),A=(1),SP=(3) Free storage before processing next profile
* B GENLOOP Process next generic profile
* Return to caller
DONE EQU * Return to caller
L 13,SAVEAREA+4 Caller's save area address
FREEMAIN R,LV=DYNLEN,A=(11) Get dynamic storage
LM 14,12,12(13) Pop registers
SLR 15,15 Clear return code
BR 14 Return to caller
* Static RACROUTE area
STATRACR RACROUTE REQUEST=EXTRACT,TYPE=EXTRACTN,ENTITYX=--*,
FIELDS=--*,SEGMENT='BASE',CLASS='TAPEVOL',
GENERIC=ASIS,WORKA=--*,RELEASE=1.9,MF=L
RACRLN EQU *-STATRACR Length of RACROUTE
* Constants
H0 DC H'0'
H6 DC H'6'
BLNKNAME DC CL6' '
* FIELDS DC A(1)
  DC CL8'OWNER'
* Result area mapping
* IRPRXTW
* Dynamic area
* DYNAREA DSECT
* DYNRACR RACROUTE REQUEST=EXTRACT,TYPE=EXTRACTN,ENTITYX=--*,
FIELDS=--*,SEGMENT='BASE',CLASS='TAPEVOL',
GENERIC=ASIS,WORKA=--*,RELEASE=1.9,MF=L
* ENTITYX structure
ENTXBUFF DS 0CL10 ENTITYX structure
ENTXLEN DS H Entity name buffer length
ENTXNLEN DS H Entity name actual length
ENTXNAME DS CL6 Entity name
* Work and save areas
WORKAREA DS 128F Work area
SAVEAREA DC 18F'0' Save area
* DYNLEN EQU *-DYNAREA Dynamic area length
* END
REQUEST=EXTRACT

Example 4

The following is an example of a RACROUTE REQUEST=EXTRACT with EXTRACTN. It retrieves all profiles within the DATASET class. The database needs to be processed only once to extract all discrete and generic profiles in the DATASET class.

EXTRTNDS CSECT
*
* Entry Linkage
*
STM 14,12,12(13) Push caller registers
BALR 12,0 Establish ...
USING *,12 ... addressability
GETMAIN R,LV=DYNLEN Get dynamic storage
LR 11,1 Move getmained address to R11
USING DYNAREA,11 Addressability to DSECT
ST 13,SAVEAREA+4 Save caller save area address
LA 15,SAVEAREA Get address of own save area
ST 15,8(13) Store in caller save area
LR 13,15 Get address of own save area
*
* Initialize variables in dynamic storage area
*
*
MVC ENTXBLEN,H44 Set buffer length to 44
MVC ENTXNLEN,H0 Set entity length to 0
MVC ENTXNAME,BLNKNAME Set entity name to blanks
*
* Copy static RACROUTE to dynamic GETMAINed areas
*
* MVC DYNRACR(RACRLEN),STATRACR
*
* Loop to retrieve the OWNER field from all DATASET profiles for each high level qualifier. Generic profiles are retrieved first.
*
* LOOP EQU * Start of loop
* RACROUTE REQUEST=EXTRACT,TYPE=EXTRACTN,ENTITYX=ENTXBUFF,
* FIELDS=FIELDS,WORKA=WORKAREA,RELEASE=1.9,MF=(E,DYNRACR)
* LTR 15,15 Check return code
* BNZ DONE Exit on nonzero return code
*
* Do DATASET profile processing here.
*
* Free storage for this profile
*
USING EXTWKEA,1 Base the result area on register 1
*
MVC DYNGENRC,EXTFLAG Make a local copy of generic flag
*
XR 3,3 Zero out register 3
ICM 3,1,EXTWSP Move the result area subpool into register 3
*
XR 2,2 Zero out register 2
ICM 2,7,EXTWLN Move the result area length into register 2
*
DROP 1 Drop basing on register 1
FREEMAIN R,LV=(2),A=(1),SP=(3) Free storage before processing next profile
*
TM DYNGENRC,X'80' Check generic bit
BO GENERIC Branch if generic bit is on
* Otherwise, profile is not generic, so set GENERIC to ASIS
REQUEST=EXTRACT

RACROUTE REQUEST=EXTRACT,TYPE=EXTRACTN,GENERIC=ASIS,
RELEASE=1.9,MF=(M,DYNRACR)

* B LOOP Process next profile

* GENERIC EQU * Profile name is generic, so set GENERIC to YES
* RACROUTE REQUEST=EXTRACT,TYPE=EXTRACTN,GENERIC=YES,
  RELEASE=1.9,MF=(M,DYNRACR)

* B LOOP Process next profile

* Return to caller

DONE EQU * Return to caller
L 13,SAVEAREA+4 Caller's save area address
FREEMAIN R,LV=DYNLEN,A=(11) Get dynamic storage
LM 14,12,12(13) Pop registers
SLR 15,15 Clear return code
BR 14 Return to caller

* Static RACROUTE area

* STATRACR RACROUTE REQUEST=EXTRACT,TYPE=EXTRACTN,ENTITYX=-*, *
  FIELDS=-*,SEGMENT='BASE',CLASS='DATASET', *
  GENERIC=ASIS,WORKA=-*,RELEASE=1.9,MF=L
RACRLEN EQU -*-STATRACR Length of RACROUTE

* Constants

* H0 DC H'0'
H44 DC H'44'
BLNKNAME DC CL44' '
* FIELDS DC A(1)
  DC CL8'OWNER'
* Result area mapping
  * IRRPRXTW
  * Dynamic area
  * DYNAREA DSECT
  * DYNRACR RACROUTE REQUEST=EXTRACT,TYPE=EXTRACTN,ENTITYX=-*, *
    FIELDS=-*,SEGMENT='BASE',CLASS='DATASET', *
    GENERIC=ASIS,WORKA=-*,RELEASE=1.9,MF=L
  * ENTITYX structure
  * ENTXBUFF DS 0CL48 ENTITYX structure
  ENTXBLEN DS H Entity name buffer length
  ENTXNLEN DS H Entity name actual length
  ENTXNAME DS CL44 Entity name
  * Work and save areas
  * WORKAREA DS 128F Work area
  SAVEAREA DC CL8'0' Save area
  DYNGENRC DS CL1 Local copy of generic flag
  * DYNLEN EQU -*DYNAREA Dynamic area length
  * END
RACROUTE REQUEST=EXTRACT (list form)

The list form of the RACROUTE REQUEST=EXTRACT macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=EXTRACT macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

| name | name: Symbol. Begin name in column 1. |
| b    | One or more blanks must precede RACROUTE. |
| RACROUTE | |
| b    | One or more blanks must follow RACROUTE. |

REQUEST=EXTRACT

,TYPE=EXTRACT
,TYPE=EXTRACTN
,TYPE=REPLACE
,TYPE=ENCRYPT
,TYPE=ENVRXTR

,ACEE=acee addr
   acee addr: A-type address

,ENTITY=profile name addr
   profile name addr: A-type address

,ENTITYX=extended profile name addr
   extended profile name addr: A-type address

,FLDACC=YES
,FLDACC=NO
   Default: FLDACC=NO

,GENERIC=ASIS
,GENERIC=YES
   Default: GENERIC=ASIS

,VOLSER=vol addr
   vol addr: A-type address

,MF=L

If TYPE=EXTRACT or EXTRACTN is specified:

,CLASS='class name'
   class name: 1–8 character name
   class name addr: A-type address
   Default: CLASS='USER'

,DATEFMT=YYYYDDDF
   Default: DATEFMT=YYDDDF

,DERIVE=YES
   See explanation of keyword.
   Default: Normal processing

,FIELDS=field addr
   field addr: A-type address

,SEGMENT='segment name'
   segment name: 1–8 character name
   segment name addr: A-type address
REQUEST=EXTRACT

name addr

,SUBPOOL=subpool number
    subpool number: Decimal digit 0–255
    Default: See explanation for SUBPOOL keyword on page 114

If TYPE=REPLACE is specified:

,BRANCH=YES
,BRANCH=NO
    Default: BRANCH=NO

,CLASS='class name'
,CLASS=class name addr
    class name: 1–8 character name
    class name addr: A-type address
    Default: CLASS='USER'

,DATEFMT=YYYYDDDF
,DATEFMT=YYDDDF
    Default: DATEFMT=YYDDDF

,FIELDS=field addr
    field addr: A-type address

,MATCHGN=YES
,MATCHGN=NO
    Default: MATCHGN=NO

,SEGDATA=segment data addr
    segment data addr: A-type address

,SEGMENT='segment name'
,SEGMENT=segment name addr
    segment name: 1–8 character name
    segment name addr: A-type address

If TYPE=ENCRYPT is specified:

,BRANCH=YES
,BRANCH=NO
    Default: BRANCH=NO

,ENCRYPT=(data data addr,DES)
,ENCRYPT=(data data addr,INST)
,ENCRYPT=(data data addr,STDDES)

,ENCRYPT=(data data addr,HASH)

If TYPE=ENVRXTR is specified:

,ENVROUT=envr data
    envr data addr: A-type address or register (2) – (12)

Note: When TYPE=ENVRXTR is specified, only the keywords ACEE and ENVROUT are recognized, with ENVROUT being required.

The parameters are explained under the standard form of the RACROUTE REQUEST=EXTRACT macro with the following exception:

,MF=L
    specifies the list form of the RACROUTE REQUEST=EXTRACT macro.
**RACROUTE REQUEST=EXTRACT (execute form)**

The execute form of the RACROUTE REQUEST=EXTRACT macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=EXTRACT macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Symbol. Begin name in column 1.</td>
</tr>
<tr>
<td></td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
<tr>
<td>RACROUTE</td>
<td>One or more blanks must follow RACROUTE.</td>
</tr>
</tbody>
</table>

```plaintext
REQUEST=EXTRACT
,TYPE=EXTRACT
,TYPE=EXTRACTN
,TYPE=REPLACE
,TYPE=ENCRYPT
,TYPE=ENVRXTR
,ACEE=acee addr
,ENTITY=profile name addr
,ENTITYX=extended profile name addr
,FLDACC=YES
,FLDACC=NO
,GENERIC=ASIS
,GENERIC=YES
,RELEASE=number
,RELEASE=(,CHECK)
,RELEASE=number,CHECK
,VOLSER=vol addr
,MF=(E,ctrl addr)

If TYPE=EXTRACT or EXTRACTN is specified:

,CLASS=class name addr
,DATEFMT=YYYYDDDF
,DATEFMT=YYDDDF
,DERIVE=YES
```

See explanation of keyword.
REQUEST=EXTRACT

.field:field addr Rx-type address or register (2) – (12)

.MATCHGN=YES

.MATCHGN=NO

.segment:segment name addr Rx-type address or register (2) – (12)

.subpool:subpool number Decimal digit 0–255

If TYPE=REPLACE is specified:

.class:class name addr Rx-type address or Register (2) – (12)

.DATEFMT=YYYYDDDF

.DATEFMT=YYDDDF Default: DATEFMT=YYDDDF

.field:field addr Rx-type address or register (2) – (12)

.segment:segment data addr Rx-type address or register (2) – (12)

.segment:segment name addr Rx-type address or register (2) – (12)

If TYPE=ENCRYPT is specified:

.BRANCH=YES

.BRANCH=NO

.data:(data addr,DES) A-type address

.data:(data addr,HASH)

.data:(data addr,INST)

.data:(data addr,STDDES)

If TYPE=ENVRXTR is specified:

.envr:envr data addr A-type address or register (2) – (12)

Note: When TYPE=ENVRXTR is specified, only the keywords ACEE and ENVROUT are recognized, with ENVROUT being required.

The parameters are explained under the standard form of the RACROUTE REQUEST=EXTRACT macro with the following exceptions:

.MF=(E,ctrl addr) specifies the execute form of the RACROUTE REQUEST=EXTRACT macro, using a remote, control-program parameter list.
REQUEST=EXTRACT

,RELEASE=number
,RELEASE=(,CHECK)
,RELEASE=(number,CHECK)

specifies the RACF release level of the parameter list to be generated by this macro.

Certain parameters can be specified only with particular releases. If you specify a parameter with an incompatible release level, the parameter is not accepted by macro processing. An error message is issued at assembly time.

The default is RELEASE=1.6.

When you specify the RELEASE keyword, checking is done at assembly time. Compatibility between the list and execute forms of the RACROUTE REQUEST=EXTRACT macro is validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.

RACROUTE REQUEST=EXTRACT (modify form)

The modify form of the RACROUTE REQUEST=EXTRACT macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=EXTRACT macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

________________________
name

: Symbol. Begin name in column 1.

b

One or more blanks must precede RACROUTE.

RACROUTE

b

One or more blanks must follow RACROUTE.

________________________
REQUEST=EXTRACT

,TYPE=EXTRACT
,TYPE=EXTRACTN
,TYPE=REPLACE
,TYPE=ENCRYPT
,TYPE=ENVRXTR

,ACEE=acee addr

acee addr: Rx-type address or register (2) – (12)

,ENTITY=profile name addr

profile name addr: Rx-type address or register (2) – (12)

,ENTITYX=extended profile name addr

extended profile name addr: Rx-type address or register (2) – (12)

,FLDACC=YES
,FLDACC=NO

,GENERIC=ASIS
,GENERIC=YES
REQUEST=EXTRACT

,.MF=(M,ctrl addr)  
  
  **ctrl addr**: Rx-type address register (1), or register (2) – (12)

If TYPE=EXTRACT or EXTRACTN is specified:

,.BRANCH=YES  
,.BRANCH=NO  

,.CLASS=class name addr  
  
  **class name addr**: Rx-type address or register (2) – (12)

,.DATEFMT=YYYYDDDF  
,.DATEFMT=YYDDDF  
  
  **Default**: DATEFMT=YYDDDF

,.DERIVE=YES  
  
  See explanation of keyword.

,.FIELDS=field addr  
  
  **field addr**: Rx-type address or register (2) – (12)

,.MATCHGN=YES  
,.MATCHGN=NO  

,.SEGMENT=segment name addr  
  
  **segment name addr**: Rx-type address or register (2) – (12)

,.SUBPOOL=subpool number  
  
  **subpool number**: Decimal digit 0–255

If TYPE=REPLACE is specified:

,.CLASS=class name addr  
  
  **class name addr**: Rx-type address or register (2) – (12)

,.DATEFMT=YYYYDDDF  
,.DATEFMT=YYDDDF  
  
  **Default**: DATEFMT=YYDDDF

,.FIELDS=field addr  
  
  **field addr**: Rx-type address or register (2) – (12)

,.SEGDATA=segment data addr  
  
  **segment data addr**: Rx-type address or register (2) – (12)

,.SEGMENT=segment name addr  
  
  **segment name addr**: Rx-type address or register (2) – (12)

If TYPE=ENCRYPT is specified:

,.BRANCH=YES  
,.BRANCH=NO  

,.ENCRYPT=(data addr,DES)  
,.ENCRYPT=(data addr,HASH)  
,.ENCRYPT=(data addr,INST)  
,.ENCRYPT=(data addr,STDDES)  
,.ENVROUT=data addr  
,.INST  
,.ENVRXTR=data addr  
,.STDDES)
REQUEST=EXTRACT

If TYPE=ENVRXTR is specified:

\[ ,\text{ENVROUT}=\text{envr data} \]
\[ \text{envr data addr: A-type address or register (2) – (12)} \]

Note: When TYPE=ENVRXTR is specified, only the keywords ACEE and ENVROUT are recognized, with ENVROUT being required.

The parameters are explained under the standard form of the RACROUTE REQUEST=EXTRACT macro with the following exceptions:

\[ ,\text{MF}=(\text{M,ctrl addr}) \]

specifies the modify form of the RACROUTE REQUEST=EXTRACT macro, using a remote, control-program parameter list.

RACROUTE REQUEST=FASTAUTH: Verify access to resources

NOT programming interface information

Use of CLASS='PROGRAM' with RACROUTE REQUEST=FASTAUTH is not part of the intended programming interface. It is intended for use only by z/OS itself, and follows different rules and has different restrictions than uses of RACROUTE REQUEST=FASTAUTH with other class names. For more information, including effects on SAF exits, see Appendix C, "System authorization facility (SAF) and SAF exits," on page 383 and Appendix D, "SAF interface to an external security product," on page 391.

End of NOT programming interface information

The RACROUTE REQUEST=FASTAUTH macro is used to check a user's authorization for access to a resource. RACROUTE REQUEST=FASTAUTH verifies access to those resources whose RACF profiles have been brought into main storage by the RACROUTE REQUEST=LIST service or by SETROPTS RACLIST. If the profiles were not RACLISTed by RACROUTE REQUEST=LIST but were processed using SETROPTS RACLIST, and the caller is in supervisor state or system key (0–7), FASTAUTH uses the SETROPTS profiles RACLISTed for the authorization check. Using SETROPTS RACLIST instead of RACROUTE REQUEST=LIST to bring profiles into storage can simplify application design. However, to get optimum performance from calls to RACROUTE REQUEST=FASTAUTH, the class should be RACLISTed using RACROUTE REQUEST=LIST.

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

The RACROUTE REQUEST=FASTAUTH caller must be authorized (supervisor state or system key 0–7) for certain keywords. See the keyword descriptions for authorization requirements.

Note: The caller must not hold any locks when issuing RACROUTE REQUEST=FASTAUTH. Programs that issue RACROUTE REQUEST=FASTAUTH while holding a lock might experience unpredictable results.
REQUEST=FASTAUTH

When a nested ACEE (or ENVR object for an ACEE created with the NESTED option of RACROUTE REQUEST=VERIFY) is passed to RACROUTE REQUEST=FASTAUTH by a supervisor-state or system-key caller and the access request is for a delegated resource (a resource defined with the RACF-DELEGATED string in the APPLDATA field), FASTAUTH processing retries the access request using the nested identity (usually, a daemon) when the primary identity has insufficient access authority.

You can use the installation exits associated with RACROUTE REQUEST=FASTAUTH to make additional security checks or to instruct RACROUTE REQUEST=FASTAUTH to either accept or fail the request. You can write an application that uses RACROUTE REQUEST=LIST and RACROUTE REQUEST=FASTAUTH for authorization checking on a resource class and associated resource group that your installation defines.

RACROUTE REQUEST=FASTAUTH is SRB-compatible. When issuing RACROUTE REQUEST=FASTAUTH in SRB mode for RACLISTed classes, you must ensure that the jobstep task pointed to by the ASCBXTCB field in the target address space is active when you schedule the SRB, and remains active until the SRB completes.

**Rule:** In order to ensure that the SRB does not run after the ASCBXTCB task completes, it is necessary to enable purgeDQ to deal with that SRB. In particular:

- If using SCHEDULE, SRBPASID must be set to the ASID, and SRBPETCB must be set to the contents of ASCBXTCB.
- If using IEAMSCHD, PURGESTOKEN must be specified with the STOKEN of the space, and PTCBADDR must be specified with the contents of ASCBXTCB.

When the task terminates, the purgeDQ issued causes the SRB's resource manager termination routine (RMTR) to be driven if the SRB has not begun to run, or waits for the SRB to complete if the SRB has begun to run.

RACROUTE REQUEST=FASTAUTH can also be invoked from a cross memory environment, when in task or SRB mode. The ACEE used for authority checking must reside in the HOME address space (unless ACEEALET specifies a different address space), but that ACEE is not used to locate the profiles RACLISTed. The main ACEE in the PRIMARY address space is used to locate the profiles.

**Restriction:** When RACROUTE REQUEST=FASTAUTH is invoked in SRB mode, WHEN (PROGRAM) conditional access checking for profiles in the SERVAUTH class is bypassed.

RACROUTE REQUEST=FASTAUTH provides support for auditing. FASTAUTH determines that auditing is necessary if at least one of the following conditions is met:

- The user represented by the ACEE or ENVR object is being audited (UAUDIT option),
- The profile protecting the resource indicates that auditing is to be done (AUDIT, GLOBALAUDIT, and WARNING options),
- The pre- or post-installation exits indicate that auditing is to be done, or
- SETROPTS SECLABELAUDIT is in effect.

The type of auditing support provided is based on the value of the LOG keyword. Either a reason code indicating whether or not logging needs to be done can be returned to the caller or RACROUTE REQUEST=FASTAUTH can perform the auditing itself. Applications can determine whether the cross-memory support and
the support enabling RACROUTE REQUEST=FASTAUTH to perform the auditing itself is available by examining the flag bit RCVTXMFR in the RCVT data area.

Restrictions:

- Callers in cross-memory mode must be in primary ASC mode when they issue the RACROUTE request.
- The RACROUTE REQUEST=FASTAUTH macro executes in the addressing mode of the caller. Therefore, to access profiles that reside above 16MB, the program that issues RACROUTE REQUEST=FASTAUTH must be running in 31-bit addressing mode when it issues RACROUTE REQUEST=FASTAUTH.

RACROUTE REQUEST=FASTAUTH (standard form)

The standard form of the RACROUTE REQUEST=FASTAUTH macro instruction is written as follows. For a description of additional keywords that you can code and additional parameters that are required on the RACROUTE request, but that are not specific to this request type, see “RACROUTE (standard form)” on page 13.

Note:

RACROUTE REQUEST=FASTAUTH requires an ACEE. For most applications, the system will have created an ACEE to represent the active user. However, for special cases where no ACEE exists, the invoker must create one before invoking RACROUTE REQUEST=FASTAUTH.

The most common way to create an ACEE is to issue a RACROUTE REQUEST=VERIFY, specifying ENVIR=CREATE. After all RACROUTE invocations are complete, the invoker should issue RACROUTE REQUEST=VERIFY with ENVIR=DELETE specified, to delete the ACEE previously created.

---

name

name: Symbol. Begin name in column 1.

b

One or more blanks must precede RACROUTE.

RACROUTE

b

One or more blanks must follow RACROUTE.

REQUEST=FASTAUTH

,CLASS='class name'

class name: 1–8 character class name

,CLASS=class name addr

class name addr: A-type address or register (2) – (12)

,ENTITY=entity addr

d entity addr: A-type address or register (2) – (12)

,ENTITYX=extended profile name addr

extended profile name addr: A-type address or register (2) – (12)

,WKAREA=area addr

area addr: A-type address or register (2) – (12)

,ACEE=acee addr

acee addr: A-type address or register (2) – (12)

,ACEEALET=alet addr

alet addr: A-type address or register (2) – (12)

,APPL='applname'

applname: 1–8 character name
REQUEST=FASTAUTH

*APPL=applname addr*  
applname addr: A-type address or register (2) – (12)

*ATTR=ATTR=READ*  
*ATTR=UPDATE*  
*ATTR=CONTROL*  
*ATTR=ALTER*  
reg: Registers (2) – (12)

*AUTHCHKS=AUTHCHKS=ALL*  
*AUTHCHKS=CRITONLY*  

*CRITERIA=criteria addr*  
criteria addr: A-type address or register (2) – (12)

*ENVRIN=envr data addr*  
envr data addr: A-type address or register (2) – (12)

*INSTLN=parm list addr*  
parm list addr: A-type address or register (2) – (12)

*LOG=ASIS*  
*LOG=NOFAIL*  
*LOG=NONE*  

*LOGSTR=logstr addr*  
logstr addr: A-type address or register (2) – (12)

*MF=S*

The parameters are explained as follows:

*ACEE=acee addr*  
specifies the address of the ACEE to be used to check authorization. If the caller is in cross-memory mode, the specified ACEE must reside in the HOME address space unless the ACEEALET= keyword specifies a different address space.

To use this keyword, the calling program must not specify the ENVRIN= keyword.

See also 136.

This ACEE might also be used to locate the profile protecting the resource. The profiles brought into storage by RACROUTE REQUEST=LIST are anchored in an ACEE. The ACEE used to access the profile protecting the resource is determined as follows:

<table>
<thead>
<tr>
<th></th>
<th>Cross-memory request</th>
<th>Non-cross-memory request</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GLOBAL=NO</td>
<td>GLOBAL=YES</td>
</tr>
<tr>
<td>ACEEALET, ENVRIN, or CRITERIA is specified.</td>
<td>Main ACEE in primary address space</td>
<td>Main ACEE in primary address space</td>
</tr>
<tr>
<td>Neither ACEEALET, ENVRIN, or CRITERIA is specified.</td>
<td>1) Input ACEE 2) Task ACEE 3) Main ACEE 4) Main ACEE 5) Main ACEE</td>
<td>1) Input ACEE if system key or supervisor state 2) Task ACEE 3) Main ACEE</td>
</tr>
</tbody>
</table>

138  z/OS V2R1.0 Security Server RACROUTE Macro Reference
REQUEST=FASTAUTH

1. If the input ACEE is not specified, then RACF uses the TASK ACEE (TCBSENV) pointer in the TCB. If there is no TCB (which is the case in SRB mode), or if the TASK ACEE pointer is zero, then RACF uses the main ACEE for the address space.

2. If the FASTAUTH caller is in system key or supervisor state, RACF uses the input ACEE if specified. If not, RACF uses the TASK ACEE. If there is no TCB (which is the case in SRB mode), or if the TASK ACEE pointer is zero, RACF uses the main ACEE for the address space.

Note: If the search of the ACEE(s) determines that the class is not RACLISTed by RACROUTE REQUEST=LIST, and the caller is in supervisor state or system key (0–7), FASTAUTH uses profiles brought into storage by the SETROPTS RACLIST command.

,ACEEALET=alet addr

specifies the address of the 4-byte ALET to be used to access the ACEE specified on the ACEE= keyword. When you use A-type or RX-type notation, alet addr specifies the name of a 4-byte field that contains the ALET. When you use register notation, alet addr specifies a register that contains the address of a 4-byte field that contains the ALET. If this keyword is not specified, the ACEE is located as defined in the ACEE= keyword description.

Keyword requirements:
1. Run in supervisor state or system key (0–7).
2. Ensure that the address space identified by ACEEALET= is marked non-swappable.
3. Ensure that the specified ALET represents a valid entry in the DU-AL of the work unit or the PASN-AL of the current primary address space.
4. Specify the ACEE= keyword.
5. Specify RELEASE=2.4 or later.

,APPL='applname'

specifies the name of the application requesting the authorization checking. This information is not used for the authorization-checking process but is made available to the installation exit or exits. If an address is specified, it should point to an 8-byte area containing the application name, left-justified and padded with blanks if necessary.

,ATTR=READ

specifies the access authority the user must have to the resource profile. The following definitions apply:

READ RACF user or group can open the resource only to read.

UPDATE RACF user or group can open the resource to read or write.

CONTROL

For VSAM data sets, RACF user or group has authority equivalent to the VSAM control password.

For non-VSAM data sets and other resources, RACF user or group has UPDATE authority.
REQUEST=FASTAUTH

ALTER
   RACF user or group has total control over the resource.

For multilevel-secure environments:
1. When ATTR=READ, it is treated as a read-only request for purposes of mandatory access control (MAC) checking.
2. When ATTR=UPDATE, CONTROL, or ALTER, it is treated as a read-write request for purposes of mandatory access control (MAC) checking.
   Note that for REQUEST=AUTH, ATTR=ALTER is treated as a read-only request for purposes of mandatory access control (MAC) checking.

If a register is specified, the register must contain one of the following codes in the low-order byte of the register:

   X'02'  READ
   X'04'  UPDATE
   X'08'  CONTROL
   X'80'  ALTER

   The default is READ.

,,AUTHCHKS=ALL
,,AUTHCHKS=CRITONLY
 specifies the access checks that RACROUTE REQUEST=FASTAUTH will make. The following definitions apply:

ALL    All RACROUTE REQUEST=FASTAUTH access checks are performed. This value is the default.

CRITONLY
   The following subset of RACROUTE REQUEST=FASTAUTH checks are performed:
   • Enforcement of the rules for SETROPTS MLQUIET, when SETROPTS MLQUIET is in effect
   • Security label authorization checks, when the SECLABEL class is active
   • Security level and security category authorization checks, when the SECLABEL class is not active and the SECDATA class is active
   • Search of the conditional access list for a matching criteria as specified by the CRITERIA keyword

For more detailed information about the authorization checking performed by RACROUTE REQUEST=FASTAUTH requests, see the section in z/OS Security Server RACF Security Administrator’s Guide on “Debugging Problems in the RACF Database”.

Note: AUTHCHKS has meaning only when the CRITERIA keyword is specified. When the AUTHCHKS=CRITONLY keyword is used without assigning a value for the CRITERIA keyword, AUTHCHKS=CRITONLY is ignored.

Keyword requirements: Specify RELEASE=7730 or later.

,,CLASS='class name'
,,CLASS='class name addr'
specifies that RACF authorization checking is to be performed for a resource of
the specified class. If an address is specified, the address must point to an 8-byte field containing the class name, left-justified and padded to the right with blanks.

\texttt{,CRITERIA=\textit{criteria-area addr}}

specifies the address of a data structure containing the additional criteria used to determine if the user has access to the resource. The criteria-area data structure is a 4-byte number of criteria entries, followed by that number of entries. The number of criteria entries must be 1. Each criteria entry consists of an 8-byte criteria-name, assumed to be in upper case, a 4-byte value length, and the criteria-value. The criteria-name, padded on the right with blanks if it is less than 8-bytes, should be made up of alphabetic, numeric, and # (X'7B'), $ (X'5B'), or @ (X'7C') characters and does not contain blanks (X'40'). The criteria-value, whose length must be in the range of 1 to 235, can contain any character but should not end with trailing blanks. The case of the characters is important. They must match what the security administrator enters on the PERMIT command.

One criteria must be specified. The PERMIT command with the \texttt{WHEN(CRITERIA(...))} is used to add criteria-expressions to the conditional access list of a general resource \textit{profile}. A \textbf{complete list of supported criteria-names} can be found in z/OS Security Server RACF Command Language Reference with the description of the PERMIT command. Access to a resource can be granted when a criteria expression on the conditional access list of a covering profile matches the expression on the call to FASTAUTH.

This processing only applies to general resource classes.

\textbf{Keyword requirements:}

1. Run in supervisor state or system key (0-7).
2. Specify RELEASE=7730 or later.
3. The criteria area must be in the primary address space.

\texttt{,ENTITY=\textit{entity addr}}

specifies that RACF authorization checking is to be performed for the resource whose name is pointed to by the specified address. The resource name is a 6-byte volume serial number for CLASS=DASDVOL or CLASS=TAPEVOL. The name must be left-justified and padded with blanks. The length of all other resource names is determined from the class descriptor table.

See also [136]

\texttt{,ENTITYX=\textit{extended profile name addr}}

specifies the address of a structure that consists of two 2-byte length fields, followed by the entity name. The first 2-byte field specifies a buffer length that can be from 0 to 255 bytes. This length field refers to the length of the buffer that contains the entity name; it does not include the length of either length field. The second 2-byte field specifies the actual length of the entity name. This length field includes the length of the actual name without any trailing blanks; it does not include the length of either length field.

These two length fields can be used in several different ways. If you know the length of the entity name, you can specify 0 in the first field and the length of the entity name in the second field. Note that each byte counts when you specify the second field. The entity name you specify must match the entity name on the RACF database. If you choose to place the entity name in a buffer area, specify the length of the buffer in the first field. For the second field, do one of the following:
REQUEST=FASTAUTH

- If you know the length of the entity name, specify the length in the second field. The length of the first field can be from 0 to 255, but must be equal to or greater than the length of the second field.
- If you do not know the length of the entity name, specify 0 in the second field, and have RACF determine the number of characters in the entity name. In this case, the entity name must be bounded on the right by a blank, unless its length is the length of the buffer itself.
- Use of ENTITYX requires RELEASE=2.6 or later.

See also 136

,ENVRIENV=envr data addr
specifies the data structure that contains the information necessary to re-create the security environment to be used for authorization checking.

The data structure describes the storage location for the ENVR object. While the format of the data structure pointed to by the ENVRIENV keyword is known to the RACROUTE invokers, the content of the ENVR object is determined by the external security product. Both the data structure, and the storage location for the ENVR object are expected to be in the PRIMARY address space.

When the ENVRIENV keyword is used, in-storage profiles RACLISTed by RACROUTE REQUEST=LIST are located using the ACEE of the address space in the primary address space. If no RACROUTE REQUEST=LIST was done for the requested class, but the class was processed using SETROPTS RACLIST, then those profiles are used.

To use this keyword the calling program must:
1. Run in supervisor state or system key (0–7)
2. Not specify the ACEE= keyword
3. Specify RELEASE=2.6 or later

The data structure that the ENVRIENV keyword points to is defined in the keyword description for ENVROUT on the RACROUTE REQUEST=VERIFY macro.

See also 136

,INSTLNSpecifies the address of an area that contains information for the RACROUTE REQUEST=FASTAUTH installation exit. This address is passed to the exit routine when the exit is given control. The INSTLN parameter is used by application or installation programs to pass information to the RACROUTE REQUEST=FASTAUTH installation exit.

,LOG=ASISAudit attempt auditing if its authorization check results in success (RC=0) or failure (RC=8), and determines whether auditing is necessary based on the following conditions:
- The user's UAUDIT setting
- The AUDIT, GLOBALAUDIT, and WARNING options in effect for the resource
- If SETR SECLABELAUDIT is in effect, then the AUDIT options in the resource SECLABEL profile

,LOG=NOFAIL
,LOG=NONE
specifies the types of access attempts to be audited. To use this keyword, you must also specify RELEASE=2.1 or later.
REQUEST=FASTAUTH

- The pre- or postprocessing installation exit's indication of whether or not to do auditing.

NOFAIL
If the authorization check fails, the attempt is not recorded. If the authorization check succeeds, the attempt is recorded as in ASIS.

NONE
The attempt is not recorded.

LOG=NONE suppresses both messages and SMF records regardless of MSGSUPP=NO.

Note: For LOG=NOFAIL and LOG=NONE, the caller should examine the reason code returned in SAFPRREA to determine whether auditing is required. If so, it is the caller's responsibility to ensure that auditing occurs, either by issuing RACROUTE REQUEST=AUTH with LOG=ASIS or RACROUTE REQUEST=FASTAUTH with LOG=ASIS.

,LOGSTR=logstr addr
specifies a variable length data string consisting of a 1-byte, binary length field followed by character data that is to be included in the RACF SMF process records. The character data can be 0–255 bytes long. When you use A-type or RX-type notation, logstr addr specifies the field name of the data string. When you use register notation, logstr addr specifies a register that contains the address of a 4-byte field that contains the data string. To use this keyword, you must also specify RELEASE=2.4 or a later release number.

,WKAREA=area addr
specifies the address of a 16-word work area to be used by RACROUTE REQUEST=FASTAUTH. On return from RACROUTE REQUEST=FASTAUTH, the following fields contain:

Word 11
The high order bit (bit 0) indicates whether a nested ACEE was used in the access decision.

Word 12
The RACF reason code.

Word 13
The RACF return code.

Word 14
The address of the in-storage profile used to determine authorization; if no profile was found or the profiles were RACLISTed using RACROUTE REQUEST=LIST, GLOBAL=YES, or SETROPTS RACLIST, word 14 contains zero. If a profile was found and not RACLISTed using RACROUTE REQUEST=LIST, GLOBAL=YES, or SETROPTS RACLIST, the profile address is passed to the caller in register 1. Note that the profile can be mapped using the RACRPE structure found in the ICHPISP macro.

Word 15
A value provided by a pre- or postprocessing installation exit, or zero.

MF=S
specifies the standard form of the RACROUTE REQUEST=FASTAUTH macro instruction.
REQUEST=FASTAUTH

Return codes and reason codes
When you execute the macro, space for the RACF return code and reason code is reserved in the first two words of the RACROUTE parameter list. You can access them using the ICHSAFP mapping macro, by loading the ICHSAFP pointer with the label that you specified on the list form of the macro. When control is returned, register 15 contains the SAF return code and register 1 might contain the address of the profile protecting the resource.

<table>
<thead>
<tr>
<th>SAF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACROUTE REQUEST=FASTAUTH has completed successfully.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>The user or group is authorized to use the resource.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>The invoker does not need to log the attempt.</td>
</tr>
<tr>
<td>04</td>
<td>The invoker should log the attempt.</td>
</tr>
</tbody>
</table>

The RACROUTE REQUEST=FASTAUTH caller should log the attempt using RACROUTE REQUEST=AUTH or RACROUTE REQUEST=FASTAUTH with LOG=ASIS.

<table>
<thead>
<tr>
<th>04</th>
<th>The requested function could not be performed.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>No security decision could be made.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACF was not called to process the request because one of the following occurred:</td>
</tr>
<tr>
<td></td>
<td>• RACF is not installed.</td>
</tr>
<tr>
<td></td>
<td>• The combination of class, REQSTOR, and SUBSYS was found in the RACF router table, and ACTION=None was specified.</td>
</tr>
<tr>
<td></td>
<td>• The RACROUTE issuer specified DECOUPL=YES and a RELEASE= keyword with a higher release than is supported by this level of z/OS.</td>
</tr>
</tbody>
</table>

| 04      | The resource or class name is not defined to RACF or the class has not been RACLSTed. |
| 0C      | RACF is not active.                                                         |
| 1C      | The class was RACLSTed by RACROUTE REQUEST=LIST, GLOBAL=YES, or SETROPTS RACLST, but the data space has been deleted. |
REQUEST=FASTAUTH

20 The class was RACLISTed by RACROUTE REQUEST=LIST, GLOBAL=YES, or SETROPTS RACLIST, but the data space cannot be accessed due to an ALESERV failure.

08 The requested function failed.

RACF RC

Meaning

Reason Code

Meaning

08 The user or group is not authorized to use the resource.

00 The invoker does not need to log the request.

04 The invoker should log the attempt.

The RACROUTE REQUEST=FASTAUTH caller should log the attempt using RACROUTE REQUEST=AUTH or RACROUTE REQUEST=FASTAUTH with LOG=ASIS.

10 A RACROUTE REQUEST=FASTAUTH installation exit error occurred.

18 Indicates the profile has a conditional access list, the port-of-entry field in the security token is blank-filled, and the port-of-entry class is active.

24 Parameter list error.

Reason Code

Meaning

8 The ACEEALET= keyword was specified, but the calling program is not running in supervisor state or system key.

0C The ACEEALET= keyword was specified, but the ACEE= keyword was not specified.

10 The ENVRIN keyword was specified, but the calling program is not running in supervisor state or system key.

14 ENVRIN and ACEE were both specified (they are mutually exclusive keywords).

18 The CRITERIA keyword was specified, but the caller was not in supervisor state or system key (0-7).

64 Indicates that the CHECK subparameter of the RELEASE keyword was specified on the execute form of the RACROUTE REQUEST=FASTAUTH macro, however, the list form of the macro does not have the same RELEASE parameter. Macro processing terminates.

Class descriptor table (CDT) default return codes and reason codes

Normally, if a resource profile is not found, the function returns a return code of 4. However, if a resource profile is not found but a default return-code keyword is specified in the class descriptor table for the class specified on the RACROUTE REQUEST=FASTAUTH, the function returns that specified return code.
**REQUEST=FASTAUTH**

### RACROUTE REQUEST=FASTAUTH (list form)

The list form of the RACROUTE REQUEST=FASTAUTH macro instruction is written as follows. Refer to the Standard Form of the RACROUTE REQUEST=FASTAUTH macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the Execute form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
</tbody>
</table>

**RACROUTE**

| b         | One or more blanks must follow RACROUTE. |

**REQUEST=FASTAUTH**

| ,ACEE=acee addr | acee addr: A-type address |
| ,ACEEALET=alet addr | alet addr: A-type address |
| ,APPL='applname' | applname: 1–8 character name |
| ,APPL=applname addr | applname addr: A-type address |
| ,ATTR=READ | Default: ATTR=READ |
| ,ATTR=UPDATE |
| ,ATTR=CONTROL |
| ,ATTR=ALTER |
| ,AUTHCHKS=ALL | Default: AUTHCHKS=ALL |
| ,AUTHCHKS=CRIPTONLY |
| ,CLASS='class name' | class name: 1–8 character class name |
| ,CLASS=class name addr | class name addr: A-type address |
| ,CRITERIA=criteria addr | criteria addr: A-type address |
| ,ENTITY=entity addr | entity addr: A-type address |
| ,ENTITYX=extended profile name addr | extended profile name addr: A-type address |
| ,ENVIRN=envr data addr | envr data addr: A-type address |
| ,INSTLN=parm list addr | parm list addr: A-type address |
| ,LOG=ASIS | Default: LOG=NONE |
| ,LOG=NOFAIL |
| ,LOG=NONE |
| ,LOGSTR=logstr addr | logstr addr: A-type address |
| ,WKAREA=area addr | area addr: A-type address |
| ,MF=L |

---

**z/OS V2R1.0 Security Server RACROUTE Macro Reference**
REQUEST=FASTAUTH

The parameters are explained under the standard form of the RACROUTE REQUEST=FASTAUTH macro instruction with the following exception:

_/MF=L_

specifies the list form of the RACROUTE REQUEST=FASTAUTH macro instruction.

**RACROUTE REQUEST=FASTAUTH (execute form)**

The execute form of the RACROUTE REQUEST=FASTAUTH macro instruction is written as follows. Refer to the Standard Form of the RACROUTE REQUEST=FASTAUTH macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the Execute form of the macro.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Symbol. Begin name in column 1.</td>
</tr>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
<tr>
<td>RACROUTE</td>
<td>One or more blanks must follow RACROUTE.</td>
</tr>
<tr>
<td>REQUEST=FASTAUTH</td>
<td></td>
</tr>
<tr>
<td>,ACEE=acee addr</td>
<td>acee addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,ACEEALET=alet addr</td>
<td>alet addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,APPL=applname addr</td>
<td>applname addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,ATTR=READ</td>
<td></td>
</tr>
<tr>
<td>,ATTR=UPDATE</td>
<td></td>
</tr>
<tr>
<td>,ATTR=CONTROL</td>
<td></td>
</tr>
<tr>
<td>,ATTR=ALTER</td>
<td></td>
</tr>
<tr>
<td>,ATTR=reg</td>
<td>reg: Register (2) – (12)</td>
</tr>
<tr>
<td>,AUTHCHKS=ALL</td>
<td></td>
</tr>
<tr>
<td>,AUTHCHKS=CRTONLY</td>
<td></td>
</tr>
<tr>
<td>,CLASS=class name addr</td>
<td>class name addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,CRITERIA=criteria addr</td>
<td>criteria addr: Rx-type address or register (2) - (12)</td>
</tr>
<tr>
<td>,ENTITY=entity addr</td>
<td>entity addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,ENTITYX=extended profile name addr</td>
<td>extended profile name addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,ENVIRN=envr data addr</td>
<td>envr data addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,INSTLN=parm list addr</td>
<td>parm list addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,LOG=ASIS</td>
<td>Default: LOG=NONE</td>
</tr>
<tr>
<td>,LOG=NOFAIL</td>
<td></td>
</tr>
<tr>
<td>,LOG=NONE</td>
<td></td>
</tr>
<tr>
<td>,LOGSTR=logstr addr</td>
<td>logstr addr: Rx-type address or register (2)–(12)</td>
</tr>
</tbody>
</table>
REQUEST=FASTAUTH

,.RELEASE=number  number: See Standard Form

,.RELEASE=(,CHECK)  Default: RELEASE=1.6
,.RELEASE=(number,CHECK)

,.WKAREA=area addr  area addr: Rx-type address or register (2) – (12)

,.MF=(E,ctrl addr)  ctrl addr: Rx-type address or register (1) – (12)

The parameters are explained under the standard form of the RACROUTE REQUEST=FASTAUTH macro instruction with the following exceptions:

,.MF=(E,ctrl addr)
   specifies the execute form of the RACROUTE REQUEST=FASTAUTH macro instruction, using a remote, control-program parameter list.

,.RELEASE=number
,.RELEASE=(,CHECK)
,.RELEASE=(number,CHECK)
   specifies the RACF release level of the parameter list to be generated by the macro.

   To use the parameters associated with a release, you must specify the number of that release or a later release number. If you specify an earlier release level, the parameter is not accepted by macro processing, and an error message is issued at assembly time.

   When you specify the RELEASE keyword, checking is done at assembly time. Compatibility between the list and execute forms of the RACROUTE REQUEST=FASTAUTH macro are validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

   The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.

RACROUTE REQUEST=LIST: Build in-storage profiles

RACROUTE REQUEST=LIST builds in-storage profiles for RACF-defined resources. RACROUTE REQUEST=LIST processes only those resources described by class descriptors. Profiles built by RACROUTE REQUEST=LIST can be used by RACROUTE REQUEST=FASTAUTH, or by RACROUTE REQUEST=AUTH, to verify a user’s access to a resource.

In-storage profiles are built from a profile on the RACF database or from potentially several profiles on the database if resource member and grouping class profiles are utilized. Each resulting in-storage profile has a size limit of 65,535 bytes. See the section in the z/OS Security Server RACF Security Administrator’s Guide on “Limiting the Size of Your Access Lists” for a discussion on access lists and other contributors to the size of an in-storage profile.

The default is RELEASE=1.6.

The module calling the RACROUTE REQUEST=LIST macro must be one of the following:
REQUEST=LIST

- Authorized (APF-authorized, in system key 0–7, or in supervisor state), or
- Link-edited with the RENT (reentrant) option, and listed in the RACF authorized caller table.

**Note:** It is recommended that if you run programs that issue the RACROUTE REQUEST=LIST macro, you run those programs APF-authorized. See z/OS Security Server RACF System Programmer’s Guide for information on the authorized caller table.

If the ACEE is below 16MB, any area chained off an ACEE, with the exception of generic profiles, is also placed below 16MB. Otherwise, the area is placed above 16MB. However, a caller executing in 31-bit mode might issue a REQUEST=LIST with LOC=ABOVE to have the profiles placed above 16MB if possible, even if the ACEE is below 16MB.

The caller cannot hold any locks when issuing RACROUTE REQUEST=LIST.

### RACROUTE REQUEST=LIST (standard form)

The standard form of the RACROUTE REQUEST=LIST macro is written as follows. For a description of additional keywords that you can code and additional parameters the are required on the RACROUTE request, but that are not specific to this request type, see “RACROUTE (standard form)” on page 13.

**Note:**

RACROUTE REQUEST=LIST requires an ACEE. For most applications, the system will have created an ACEE to represent the active user. However, for special cases where no ACEE exists, the invoker must create one before invoking RACROUTE REQUEST=LIST.

The most common way to create an ACEE is to issue a RACROUTE REQUEST=VERIFY, specifying ENVIR=CREATE. After all RACROUTE invocations are complete, the invoker should issue RACROUTE REQUEST=VERIFY with ENVIR=DELETE specified, to delete the ACEE previously created.

Application programs must be structured so that a task requesting RACF services does not do so while other I/O initiated by the task is outstanding. If such I/O is required, the task should either wait for the other I/O to complete before requesting RACF services, or the other I/O should be initiated under a separate task. This is necessary to assure proper processing in recovery situations.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
<tr>
<td>RACROUTE</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>One or more blanks must follow RACROUTE.</td>
</tr>
</tbody>
</table>

| REQUEST=LIST |
| 'CLASS='class name' | class name: 1–8 character name |
| ,CLASS=class name addr | class name addr: A-type address or register (2) – (12) |

Chapter 3. System macros 149
REQUEST=LIST

,ACEE=acee addr
     acee addr: A-type address or register (2) – (12)

,APPL='applname'
     applname: 1–8 character name

,APPL=applname addr
     applname addr: A-type address or register (2) – (12)

,ENVIR=CREATE
     Default: ENVIR=CREATE

,ENVIR=DELETE

,FILTER=filter addr
     filter addr: A-type address or register (2) – (12)

,GLOBAL=YES
     Default: GLOBAL=NO

,GLOBAL=NO

,INSTLN=parm list addr
     parm list addr: A-type address or register (2) – (12)

,LIST=list addr
     list addr: A-type address or register (2) – (12)

,LOC=ANY
     Default: See parameter description.

,LOC=ABOVE

,OWNER=YES
     Default: OWNER=NO

,OWNER=NO

,SUBPOOL=(sub#1,sub#2)
     sub#1,sub#2: Decimal digit 0–255

,MF=S

The parameters are explained as follows:

,ACEE=acee addr
     specifies the address of the ACEE. RACF uses the ACEE to anchor the list of in-storage profiles.

     If an ACEE is not specified, RACF uses the TASK ACEE pointer in the extended TCB called the TCBSENV. If the TASK ACEE pointer is zero, RACF uses the main ACEE (pointed to by the ASXBSENV field of the address-space extension block) to anchor the list of the in-storage profiles. If there is no main ACEE, the in-storage profiles are not constructed.

,APPL='applname'

,APPL=applname addr
     specifies the name of the application requesting the authorization-checking. This information is not used for the authorization-checking process but is made available to the installation exit or exits. If an address is specified, it should point to an 8-byte area containing the application name, left-justified and padded with blanks if necessary.

,CLASS='class name'

,CLASS=class name addr
     specifies that RACROUTE REQUEST=LIST is to build an in-storage profile for the resources of the specified class. If an address is specified, the address must point to an 8-byte field containing the class name, left-justified and padded with blanks, if necessary.

     The class name must be a valid, active class as defined in the class descriptor table. It must also be a member class. If the member class has a grouping class associated with it, RACF utilizes both the member and the grouping class when building the in-storage profiles.
REQUEST=LIST

,ENVIR=CREATE
,ENVIR=DELETE

specifies the action to be performed by the RACROUTE REQUEST=LIST macro.

CREATE
In-storage profiles for the specified class are to be built. The RACROUTE REQUEST=LIST function issues a return code of 18 if an in-storage list currently exists for the specified class.

DELETE
The in-storage profiles for the specified class are to be freed. If class is not specified, the in-storage profiles for all classes are freed.

Note:
1. The user issuing the RACROUTE REQUEST=LIST macro has the responsibility to ensure that no multitasking that results in the issuing of a RACROUTE REQUEST=AUTH, RACROUTE REQUEST=FASTAUTH, RACROUTE REQUEST=VERIFY, or RACROUTE REQUEST=LIST macro occurs at the same time as the RACROUTE REQUEST=LIST.
2. When a user has issued a RACROUTE REQUEST=LIST, ENVIR=CREATE to build in-storage profiles, it is the user’s responsibility to issue a RACROUTE REQUEST=LIST, ENVIR=DELETE to delete the in-storage profiles when they are no longer needed. Failure to do so can cause unpredictable results.
3. When profiles have been RACTIONed with GLOBAL=YES, ENVIR=DELETE removes the user’s access to the data space. The data space is not altered. The data space can be deleted by a SETROPTS NORACLST (classname) command, but the command should not be issued until all of the applications that accessed the data space using RACROUTE REQUEST=LIST, ENVIR=CREATE, GLOBAL=YES request have relinquished their access by issuing RACROUTE REQUEST=LIST, ENVIR=DELETE requests. The classes processed by a SETROPTS NORACLST command include not only the class specified on the command, but all valid classes that share the same POSIT.

,FILTER=filter addr

specifies the address of a generic filter string, which RACF uses to search the RACF database and select profile names for which RACROUTE REQUEST=LIST builds in-storage profiles. The filter consists of a 2-byte length field followed by the filter string. The filter-string length must not exceed the length of the profile name as it is specified in the class descriptor table.

Generic characters have special meaning when used as part of the filter string. Even when profiles do not allow an asterisk (*) in the high-level qualifier, the FILTER operand does allow it.

• % (character in a name)
  You can use the percent sign to represent any one character in the profile name, including a generic character. For example, if you specify DASD% as a filter string, it can represent profile names such as DASD01, DASD02, and DASD%. If you specify %%%% as a filter string, it can represent profile names such as DASD1, DASD2, DASD%, TAPE%, MY%%%%, TAPE* and %%%%.

• * (0 through n characters in a qualifier)
  You can use a single asterisk to represent zero or more characters in a qualifier, including generic characters. For example, AB*.CD can represent profile names such as AB.CD, ABEF.CD, and ABX.CD. A single asterisk can also
**REQUEST=LIST**

represents an entire qualifier. For example, ABC.* represents profile names such as ABC.D, ABC.DE, ABC.%%% and ABC.%/DE.

- ** (0 through n qualifiers in a name)
  
  You can use a double asterisk to represent zero or more qualifiers in the profile name. For example, AB.**.CD represents profile names such as AB.CD, AB.DE, EF.CD, and AB.XYZ.CD. You cannot specify other characters with ** within a qualifier. For example, you can specify USER1.** but not USER1.A**.

**Note:**

1. To specify the filter function, you must also specify RELEASE=1.9 or a later release number.
2. You cannot specify FILTER with LIST on the same invocation, because the two keywords are mutually exclusive.
3. You can specify the FILTER keyword with ENVIR=CREATE. If you specify ENVIR=DELETE, RACROUTE REQUEST=LIST returns a return code of 18.
4. You cannot specify the FILTER keyword with GLOBAL=YES.
5. The filter string is case sensitive. Be sure to take this into account when processing mixed-case classes (those specified in the class descriptor table with CASE=ASIS on the ICHERCDE macro, or CASE(ASIS) on the RDEFINE CDT command.)

**GLOBAL=NO**

specifies whether or not RACF uses a data space for storage of RACLISTed profiles. To use this keyword, you must also specify RELEASE=2.1 or later.

**NO** The request uses local private storage for RACLIST profile lists

**YES** Allows profiles to be shared globally in a data space. One of the following occurs:

- If profiles already exist in a data space created by SETROPTS RACLIST or by RACROUTE REQUEST=LIST,GLOBAL=YES, RACF obtains an ALET in order to use the existing data space.
- If profiles do not exist in a data space and the RACGLIST class is active, and RACF-created RACGLIST profiles exist on the RACF database, that is, classname_00000, then RACROUTE REQUEST=LIST,GLOBAL=YES creates a data space for the specified class using the RACF-created RACGLIST profiles rather than the original class profiles.
- If profiles do not exist in a data space, the RACGLIST class is active and the RACGLIST classname profile exists on the RACF database, but no RACF-created RACGLIST profiles exist, then RACF does normal RACLIST processing using the original class profiles, and copies the results into a data space. Additionally, the results of this RACLIST are copied back out to the RACF database in the RACGLIST profiles. For example, classname_00001, classname_00002.
- If profiles do not exist in a data space and the RACGLIST class is not active, or if it is active but no RACGLIST classname profile exists on the RACF database, RACF does normal RACLIST processing using the original class profiles and copies the results into a data space.

Once created, the RACLIST data space can be refreshed when appropriate by a SETROPTS RACTION REQUEST=CLASS (classname) REFRESH
command. The classes processed by this command are not only the classes specified on the command, but every valid class that shares the same POSIT as the class specified.

SETROPTS LIST can be used to determine which classes have been RACLISTed by this means. Even though the class was not RACLISTed, the fact that it was RACLISTed with GLOBAL=YES means that it has an affect across more than one application since subsequent applications issuing RACROUTE REQUEST=LIST, ENVIR=CREATE, GLOBAL=YES simply uses the already existing data space. A new line has been added to SETROPTS LIST output to show classes that have been RACLISTed solely by this means:

GLOBAL=YES RACLST ONLY =

The SETROPTS NORACLST (classname) command can be used to delete the RACLIST data space, but should not be issued until all of the applications that accessed the data space by issuing a RACROUTE REQUEST=LIST, ENVIR=CREATE, GLOBAL=YES have relinquished their access by issuing RACROUTE REQUEST=LIST ENVIR=DELETE requests. As with SETR RACLST REFRESH, the SETR NORACLST command processes not only the class specified on the command, but all the valid classes sharing the same POSIT.

Restrictions:
1. To use GLOBAL=YES, you must use RACROUTE and you must specify RELEASE=2.1 or later.
2. You cannot specify FILTER or LIST with GLOBAL=YES.
3. You can use the GLOBAL operand with ENVIR=CREATE. If you specify GLOBAL=YES with ENVIR=DELETE, RACF ignores the GLOBAL operand.
4. When GLOBAL=YES is specified, SETROPTS RACLST(classname) REFRESH must be issued by the security administrator to cause updated profiles to be loaded. The application cannot get updated profiles loaded by issuing RACROUTE REQUEST=LIST,ENVIR=DELETE followed by another RACROUTE REQUEST=LIST,ENVIR=CREATE,GLOBAL=YES. The process of refreshing by SETROPTS is transparent to the application. That is, the application can continue to issue RACROUTE REQUEST=FASTAUTH or RACROUTE REQUEST=AUTH while the refresh operation is in progress.
5. GLOBAL=YES is not supported for profiles in the CDT class.

Note:
1. GLOBAL=YES allows RACROUTE REQUEST=FASTAUTH to run in AR mode in order to access profiles in a data space.
2. Applications can determine whether GLOBAL=YES is supported by checking flag bit RCVTGLBL in the RCVT data area.

\INSTLN\parm list addr

specifies the address of an area that contains parameter information for the RACROUTE REQUEST=LIST installation exit. The address is passed to the installation exit when the exit is given control by the RACROUTE REQUEST=LIST routine. The INSTLN parameter can be used by an application or an installation program to pass information to the RACROUTE REQUEST=LIST installation exit.

\LIST=addr

specifies the address of a list of resource names for which RACROUTE REQUEST=LIST is to build the in-storage profiles. The list consists of a 2-byte field containing the number of the names in the list, followed by one or more
REQUEST=LIST

variable-length names. Each name consists of a 1-byte length field, which is the
length of the name, followed by the name. A zero in the 2-byte field causes the
operand to be omitted.

If LIST= and FILTER= are omitted, in-storage profiles are built for all the
profiles defined to RACF in the given class as well as each member for a
resource grouping associated with the specified class.

Note:
1. This operand can be specified with ENVIR=CREATE. If ENVIR=DELETE is
   specified, the RACROUTE REQUEST=LIST macro issues a return code of
   18.
2. You cannot specify the LIST keyword with GLOBAL=YES.

,LOC=ANY
,LOC=ABOVE

specifies whether the RACROUTE REQUEST=LIST profiles for GLOBAL=NO
or the class tree anchor element for GLOBAL=YES are to reside where the
ACEE is located, above or below 16MB (LOC=ANY), or whether the
RACROUTE REQUEST=LIST profiles for GLOBAL=NO or the class tree
anchor element for GLOBAL=YES are to reside above 16MB (LOC=ABOVE), if
possible, even if the ACEE is below 16MB.

Note: LOC=ANY does not guarantee that storage is allocated above 16MB. If
any installation SAF or RACF exit routines execute in 24-bit mode, the storage
is below 16MB.

,OWNER=YES
,OWNER=NO

specifies that the resource owner is to be placed in the profile access list with
the ALTER authority. If the OWNER= operand is omitted, the default is NO.

,SUBPOOL=(sub#1,sub#2)

specifies the subpool numbers of the storage into which the components of the
in-storage profiles are to be built with GLOBAL=NO. sub#1 represents the
subpool of the profile proper. sub#2 represents the subpool of the profile index.

If the subpools are not specified, they default to subpool 255. Registers can also
be used to specify sub#1 and sub#2. In that case, the low-order byte in the
register is the subpool value.

Note:
1. The subpool parameter is intended to be used when profiles are loaded
   into local storage through GLOBAL=NO. Therefore, if you use
   GLOBAL=YES to load profiles into a data space, the subpool information is
   not applicable.
2. Take some care in selecting a subpool, as MVS makes certain assumptions
   about subpool usage and characteristics. In particular, using subpool 0 or
   250, or any subpool documented in [z/OS MVS Programming: Assembler
   Services Guide] as having a storage key of USER (for example, 227–231, and
   241) can give unpredictable results.

In choosing a subpool, be aware that the storage obtained is attached to
MVS control blocks, so subpool characteristics need to be considered. For
example, storage obtained by RACROUTE REQUEST=LIST is attached to
the ACEE. If the storage for this ACEE is in subpool 255, it exists until a
RACROUTE REQUEST=VERIFY, ENVIR=DELETE is issued to delete the
ACEE. However, if the wrong subpool is chosen, the RACROUTE
REQUEST=LIST

REQUEST=LIST storage might already have been freed (for example, during step termination), so the ACEE points to storage that is no longer available.

3. If a common-area subpool (for example 226–228, 231, 239, 241, 245, 247, or 248) is used and not freed before the job terminates, then the job might show up in the exception reports of RMF (or other monitoring tools that support the tracking of common-area storage utilization) as owning common storage. Before your job terminates, it should issue a RACROUTE REQUEST=LIST, ENVIR=DELETE to free this common storage.

,MF=S

specifies the standard form of the RACROUTE REQUEST=LIST macro instruction.

Return codes and reason codes

When you execute the macro, space for the RACF return code and reason code is reserved in the first two words of the RACROUTE parameter list. You can access them using the ICHSAFP mapping macro, by loading the ICHSAFP pointer with the label that you specified on the list form of the macro. When control is returned, register 15 contains the SAF return code.

<table>
<thead>
<tr>
<th>SAF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACROUTE REQUEST=LIST completed successfully.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACROUTE REQUEST=LIST function completed successfully.</td>
</tr>
</tbody>
</table>

Reason Code

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
</tr>
<tr>
<td>04</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SAF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>The requested function could not be performed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>No security decision could be made.</td>
</tr>
</tbody>
</table>

Reason Code

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
REQUEST=LIST

- The RACROUTE issuer specified DECOUPL=YES and a RELEASE= keyword with a higher release than is supported by this level of z/OS.

08 The specified class is not defined to RACF, and the router table was bypassed due to DECOUPL=YES.

10 RACF or the resource class, or both, are not active.

14 RACLST installation-exit error occurred.

08 The requested function failed.

RACF RC
Meaning

04 Unable to perform the requested function.

Reason Code
Meaning

00 Unable to establish an ESTAE environment.

01 The function code (the third byte of the parameter list) does not represent a valid function. 01 represents the RACF manager; 02 represents the RACROUTE REQUEST=LIST macro.

0C An error was encountered during RACROUTE REQUEST=LIST processing.

18 Parameter list error.

Reason Code
Meaning

00 No ACEE found.

04 Class already RACLSTed.

08 Name length in list of names is not valid.

0C LIST or FILTER specified on DELETE request.

10 Request type is not valid (not DEFINE or DELETE).

14 LIST and FILTER specified (they are mutually exclusive).

18 LIST or FILTER specified with GLOBAL=YES.

1C Specified class not allowed with GLOBAL=YES.

1C RACF is not installed or an insufficient level of RACF is installed.

20 Filter sequence not valid.

24 REQUEST=LIST failed to return a dataspace.

Reason Code
Meaning

00 Indicates one of the following:
- attempt to RACLST regular class profiles from the RACF database failed, or
- attempt to use an existing data space was unsuccessful.

64 Indicates that the CHECK subparameter of the RELEASE keyword was specified on the execute form of the RACROUTE REQUEST=LIST macro;
However, the list form of the macro does not have the same RELEASE parameter. Macro processing terminates.

**Note:** If the resource class specified by the CLASS= operand is inactive, RACROUTE REQUEST=LIST does not build the in-storage profiles and a code of 0C is returned. If the resource group class is not active, RACROUTE REQUEST=LIST builds an in-storage profile but only from the individual resource profiles; resource-group profiles are ignored.

### Example 1
Use the standard form of the macro to build in-storage profiles for all the profiles in the APPCLU, and chain them off the ACEE whose address is pointed to by ACEEADDR.

```clike
RACROUTE REQUEST=LIST,CLASS='APPCLU',ACEE=ACEEADDR, X
ENVIR=CREATE,WORKA=RACWK
```

```clike
RACWK DS CL512
```

### Example 2
Use the standard form of the macro to build in-storage profiles for all the profiles whose names are in a list named PROFLIST and in the APPCLU class. Chain them from the task ACEE or address space ACEE.

```clike
RACROUTE REQUEST=LIST,CLASS='APPCLU',LIST=PROFLIST, X
ENVIR=CREATE,WORKA=RACWK
```

```clike
PROFLIST DS OCL58
PROFNUM DC XL2'0004'
PROF1 DC AL1(12),CL12 'NETA.LU1.LU2'
PROF2 DC AL1(12),CL12 'NETB.LU1.LU2'
PROF3 DC AL1(14),CL14 'NETONE.LUA.LUB'
PROF4 DC AL1(14),CL14 'NETTWO.LUA.LUB'
RACWK DC CL512
```

### Example 3
Use the standard form of the macro to delete the in-storage profiles for the APPCLU class.

```clike
RACROUTE REQUEST=LIST,CLASS='APPCLU',ENVIR=DELETE,WORKA=RACWK
```

```clike
RACWK DS CL512
```

### Example 4
Use the standard form of the macro to build in storage all the profiles in the specified class that match the filter string.

```clike
RACROUTE REQUEST=LIST,CLASS='APPCLU',ENVIR=CREATE X
FILTER=FILTR,RELEASE=1.9.2.WORKA=RACWK
```

```clike
FILTR DS OCL14
FILTRL DC XL2'000C'
FILTRT DC CL12 'NET*.LU*.LU*'
RACWK DS CL512
```

### Example 5
Use the standard form of the macro to load profiles into a data space, or attach to an existing data space if there is one, for the TCICSTRN class:
**REQUEST=LIST**

```
RACROUTE REQUEST=LIST,ENVIR=CREATE,CLASS='TCICSTRN',
   GLOBAL=YES,RELEASE=2.1,WORKA=RACRWRKA,MF=S
```

RACRWRKA DS 128F

**RACROUTE REQUEST=LIST (list form)**

The list form of the RACROUTE REQUEST=LIST macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=LIST macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
</tbody>
</table>

RACROUTE

| b    | One or more blanks must follow RACROUTE. |

<table>
<thead>
<tr>
<th>REQUEST=LIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>,ACEE=acee addr</td>
</tr>
<tr>
<td>,APPL='applname'</td>
</tr>
<tr>
<td>,APPL=applname addr</td>
</tr>
<tr>
<td>,CLASS='class name'</td>
</tr>
<tr>
<td>,CLASS=class name addr</td>
</tr>
<tr>
<td>,ENVIR=CREATE,ENVIR=DELETE</td>
</tr>
<tr>
<td>,FILTER=filter addr</td>
</tr>
<tr>
<td>,GLOBAL=YES,GLOBAL=NO</td>
</tr>
<tr>
<td>,INSTLN=parm list addr</td>
</tr>
<tr>
<td>,LIST=list addr</td>
</tr>
<tr>
<td>,LOC=ANY,LOC=ABOVE</td>
</tr>
<tr>
<td>,OWNER=YES,OWNER=NO</td>
</tr>
<tr>
<td>,SUBPOOL=(sub#1,sub#2)</td>
</tr>
<tr>
<td>,MF=L</td>
</tr>
</tbody>
</table>

The parameters are explained under the standard form of the RACROUTE REQUEST=LIST macro with the following exception:
**REQUEST=LIST**

.,MF=L

specifies the list form of the RACROUTE REQUEST=LIST macro instruction.

---

**RACROUTE REQUEST=LIST (execute form)**

Refer to the standard form of the RACROUTE REQUEST=LIST macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

The execute form of the RACROUTE REQUEST=LIST macro is written as follows.

```plaintext
name

name: Symbol. Begin name in column 1.

b

One or more blanks must precede RACROUTE.

RACROUTE

b

One or more blanks must follow RACROUTE.

REQUEST=LIST

.,ACEE=acee addr

acee addr: Rx-type address or register (2) – (12)

.,APPL=applname addr

applname addr: Rx-type address or register (2) – (12)

.,CLASS=class name addr

class name addr: Rx-type address or register (2) – (12)

.,ENVIR=CREATE

.,ENVIR=DELETE

.,FILTER=filter addr

filter addr: Rx-type address or register (2) – (12)

.,GLOBAL=YES

Default: GLOBAL=NO

.,GLOBAL=NO

.,INSTLN=parm list addr

parm list addr: Rx-type address or register (2) – (12)

.,LIST=list addr

list addr: Rx-type address or register (2) – (12)

.,LOC=ANY

.,LOC=ABOVE

.,OWNER=YES

.,OWNER=NO

.,RELEASE=number

number: See standard form

 Default: RELEASE=1.6

.,RELEASE=(,CHECK)

.,RELEASE=(number,CHECK)

.,SUBPOOL=(sub#1,sub#2)

sub#1,sub#2: Decimal digit 0–255

.,MF=(E,ctrl addr)

ctrl addr: Rx-type address or register (2) – (12)
```
REQUEST=LIST

The parameters are explained under the standard form of the RACROUTE REQUEST=LIST macro with the following exceptions:

,RELEASE=number
,RELEASE=(),CHECK
,RELEASE=(number,CHECK)

specifies the RACF release level of the parameter list to be generated by this macro.

To use the parameters associated with a release, you must specify the number of that release or a later release number. If you specify a parameter with an incompatible release level, the parameter is not accepted by macro processing.

When you specify the RELEASE keyword, checking is done at assembly time. Compatibility between the list and execute forms of the RACROUTE REQUEST=LIST macro is validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.

,MF=(E,ctrl addr)

specifies the execute form of the RACROUTE REQUEST=LIST macro instruction, using a remote, control-program parameter list.

RACROUTE REQUEST=LIST (modify form)

The modify form of the RACROUTE REQUEST=LIST macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=LIST macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
</tbody>
</table>

RACROUTE

b

One or more blanks must follow RACROUTE.

REQUEST=LIST

,ACEE=acee addr  acee addr: Rx-type address or register (2) – (12)
,APPL=applname addr  applname addr: Rx-type address or register (2) – (12)
,CLASS=class name addr  class name addr: Rx-type address or register (2) – (12)
,FILTER=filter addr  filter addr: Rx-type address or register (2) – (12)
,GLOBAL=YES
,GLOBAL=YES Default: GLOBAL=NO
,GLOBAL=NO
,INSTLN=parm list addr  parm list addr: Rx-type address or register (2) – (12)
,ENVIR=CREATE
The parameters are explained under the standard form of the RACROUTE REQUEST=LIST macro with the following exceptions:

\[,MF=(M,ctrl \text{ addr})\]

specifies the modify form of the RACROUTE REQUEST=LIST macro instruction, using a remote, control-program parameter list.

This form of the RACROUTE REQUEST=LIST macro requires RELEASE=1.9 or later.

**RACROUTE REQUEST=SIGNON: Manage PV signed-on lists**

The RACROUTE REQUEST=SIGNON macro is used to provide management of the signed-on lists associated with persistent verification (PV), a feature of the APPC architecture of LU 6.2. The macro is intended for use in programs which implement the APPC LU persistent verification receive architecture; it is not required in applications which merely use APPC LU 6.2 functions.

Persistent verification is logon (signon) of a user from a remote LU which persists over multiple conversations. The signed-on lists are used to manage who is signed on using PV. The RACROUTE REQUEST=SIGNON macro is the interface provided to manage signed-on-from lists on MVS. The APPL, POE, GROUP, and USER keywords are used to uniquely identify entries in the signed-on-from list(s).

The caller can use RACROUTE REQUEST=SIGNON to:

- Create signed-on-from lists explicitly (TYPE=LSTCRT) or implicitly (TYPE=SIGNIN).
- Delete signed-on-from lists (TYPE=LISTDEL).
- Add a user to a list (TYPE=SIGNIN). A user should be authenticated using RACROUTE REQUEST=VERIFY before being added to a signed-on-from list.
- Delete a user from a list (TYPE=SIGNOFF).
- Query a list (TYPE=QSIGNON) to see if a specific user is present. If the user is signed on, a copy of the user’s ACEE can be returned. In addition, the ENVROUT parameter can be used on a TYPE=QSIGNON request to obtain a relocatable user security environment that can be used on a subsequent RACROUTE REQUEST=VERIFY to provide “fastpath” user security environment (ACEE) re-creation.

One benefit of persistent verification is a reduction in the number of APPLAUDIT audit records created for a user issuing APPC transactions. Without PV, when you...
REQUEST=SIGNON

audit user verification for a user submitting APPC transactions, a pair of audit
records is written for each transaction. With PV, only two records are written for
the same user—one for the user’s initial verification, and another when the user’s
PV session ends. Refer to the description of the ENVRIN keyword parameter for
implementation details. Refer to \textit{z/OS Security Server RACF Auditor's Guide} for a
discussion of SETROPTS APPLAUDIT option.

The caller of RACROUTE REQUEST=SIGNON is required to be AMODE(31), and
must be in supervisor state and system key.

To use this service, you must specify RELEASE=1.9.2 or a later release number.

The caller cannot hold any locks.

All request types support task mode; in addition, the TYPE=LISTDEL request can
be invoked in SRB mode.

RACF might return an ACEE and/or a TOKEN and/or an ENVRUT object
storage area that resides above 16 megabytes.

\textbf{RACROUTE REQUEST=SIGNON (standard form)}

The standard form of the RACROUTE REQUEST=SIGNON macro instruction is
written as follows. For a description of additional keywords that you can code and
additional parameters that are required on the RACROUTE request, but that are
not specific to this request type, see \textit{“RACROUTE (standard form)” on page 13}.

\begin{verbatim}
Note: Application programs must be structured so that a task requesting RACF services
does not do so while other I/O initiated by the task is outstanding. If such I/O is
required, the task should either wait for the other I/O to complete before requesting
RACF services, or the other I/O should be initiated under a separate task. This is
necessary to assure proper processing in recovery situations.
\end{verbatim}

\begin{verbatim}
name

begin name in column 1.

\textbf{b}

One or more blanks must precede RACROUTE

RACROUTE

\textbf{b}

One or more blanks must follow RACROUTE

REQUEST=SIGNON

,APPL='applname'

applname: 1–8 character application name

,APPL=applname addr

applname addr: A-type address or register (2) – (12).

,POE=port of entry addr

port of entry addr: A-type address or register (2) – (12)

,POENET=network name addr

network name addr: A-type address or register (2) – (12)

,TYPE=LISTCRT

,TYPE=LISTDEL

,TYPE=SIGNIN

,TYPE=SIGNOFF

,TYPE=QSIGNON

,LSTTYPE=ONFROM

Default: LSTTYPE=ONFROM
\end{verbatim}
REQUEST=SIGNON

,MF=S

,VERBEXIT=address of fullword

exit addr: A-type address or register (2) – (12)

If TYPE=SIGNIN is specified:

,USERID=userid addr

userid addr: A-type address or register (2) – (12)

,ENVRIN=envr data addr

envr data addr: A-type address or register (2) – (12)

,GROUP=group addr

group addr: A-type address or register (2) – (12)

,SECLABL=seclabel addr

seclabel addr: A-type address or register (2) – (12)

,VERBEXIT=address of fullword

address of fullword: A-type address or register (2) – (12)

If TYPE=SIGNOFF is specified:

,USERID=userid addr

userid addr: A-type address or register (2) – (12)

,GROUP=group addr

group addr: A-type address or register (2) – (12)

,SECLABL=seclabel addr

seclabel addr: A-type address or register (2) – (12)

,VERBEXIT=address of fullword

Note: This keyword is required on or before TYPE=SIGNOFF. Refer to VERBEXIT keyword description for more information.

If TYPE=QSIGNON is specified:

,USERID=userid addr

userid addr: A-type address or register (2) – (12)

,ACEE=address of fullword

address of fullword: A-type address or register (2) – (12)

,ENVROUT=envr data addr

envr data addr: A-type address or register (2) – (12)

,GROUP=group addr

group addr: A-type address or register (2) – (12)

,SECLABL=seclabel addr

seclabel addr: A-type address or register (2) – (12)

,TOKNOUT=utoken addr

utoken addr: A-type address or register (2) – (12)

The parameters are explained as follows:

,ACEE=address of fullword

specifies the address of a fullword where a QSIGNON request is to store an ACEE pointer for a signed-on user. If the keyword is omitted (or zero), an ACEE is not returned. The caller is responsible for doing a RACROUTE REQUEST=VERIFY, ENVIR=DELETE to delete a returned ACEE when no longer needed.

,APPL='applname'
**REQUEST=SIGNON**

\texttt{APPL=applname addr}

specifies the local LU name. The local LU name is an 8-character field which is left-justified and padded with blanks.

The APPL name is part of the key that uniquely identifies which signed-on list entries are being processed.

Note that an asterisk (*) should not be specified for the APPL name. It is not treated as a generic character, and will not match the application name when used in the key.

The maximum number of APPL names that RACF supports in the signed-on lists is 39.

\texttt{ENVRIN=envr data addr}

specifies the data structure that contains the information required to re-create a security environment. The address points to a data structure defined in Table 7 on page 165. See also the mapping for “SGNPL: RACROUTE REQUEST=SIGNON Parameter List (Request Section)” in \textit{z/OS Security Server RACF Data Areas}. The data structure describes the storage location for the ENVR object. While the format of the data structure pointed to by ENVRIN is known to the RACROUTE invoker, the content of the object itself is defined by the external security product.

The ENVRIN keyword is provided to pass in a relocatable security environment or to manipulate a user security environment. You can specify ENVRIN on a RACROUTE REQUEST=SIGNON, TYPE=SIGNIN to mark a security environment so that future RACROUTE REQUEST=VERIFY, ENVIR=DELETE requests of that user security environment do not get audited.

To use ENVRIN on a RACROUTE REQUEST=SIGNON, TYPE=SIGNIN you must have previously obtained a relocatable user security environment by specifying ENVROUT on a prior service such as RACROUTE REQUEST=VERIFY, ENVIR=CREATE, or RACROUTE REQUEST=EXTRACT, TYPE=ENVRXTR.

\texttt{ENVROUT=envr data addr}

specifies the data structure that is to hold the information used to describe a security environment.

This information can be used later on a service with the ENVRIN keyword such as RACROUTE REQUEST=VERIFY to recreate the security environment without causing I/O to the RACF Database. The address points to a data structure defined in Table 7 on page 165. The data structure describes the storage location for the ENVR object. The key of the ENVR data structure is a single byte value that represents the associated ENVR object storage area. The low-order nibble of this value is the storage key. A key value of X’07’ returns an ENVR object in key 07 storage.

While the format of the data structure pointed to by ENVROUT is known to the RACROUTE invokers, the content of the object itself is defined by the external security product. Figure 4 on page 165 and Table 7 on page 165 represent the ENVR data structure for use with the ENVRIN and ENVROUT keywords. The data structure must start on a fullword boundary.
### Table 7. Description of ENVR data structure

<table>
<thead>
<tr>
<th>Description</th>
<th>Length (bytes)</th>
<th>ENVROUT usage</th>
<th>ENVRIN Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENVR object length</td>
<td>4</td>
<td>Output</td>
<td>Input</td>
</tr>
<tr>
<td>ENVR object storage area length</td>
<td>4</td>
<td>Input/Output</td>
<td>Input</td>
</tr>
<tr>
<td>ENVR object storage area address</td>
<td>4</td>
<td>Input/Output</td>
<td>Input</td>
</tr>
<tr>
<td>ENVR object storage area subpool</td>
<td>1</td>
<td>Input</td>
<td>N/A</td>
</tr>
<tr>
<td>ENVR object storage area key</td>
<td>1</td>
<td>Input</td>
<td>N/A</td>
</tr>
</tbody>
</table>

The ENVR object storage area can be supplied by the caller or obtained by RACF. If supplied by the caller, it must be on a doubleword boundary and be associated with the job step task. If RACF obtains the storage area, it is on a doubleword boundary and is associated with the Job Step task. The storage will be allocated based on the mode of the caller (LOC=ANY for 31-bit callers and LOC=BELOW for 24-bit callers). The following table shows how the field values affect ENVROUT processing.

### Table 8. ENVROUT Storage Area Processing

<table>
<thead>
<tr>
<th>ENVR object storage area length</th>
<th>ENVR object storage area address</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero</td>
<td>Any value</td>
<td>RACF obtains minimum storage size needed to contain the ENVR Object. Storage size is returned in the ENVR Object storage area length. Storage address is returned in the ENVR Object storage area address.</td>
</tr>
<tr>
<td>Nonzero</td>
<td>Zero</td>
<td>RACF obtains storage size specified or the minimum storage size needed to contain the ENVR Object. Storage size is returned in the ENVR Object storage area length. Storage address is returned in the ENVR Object storage area address.</td>
</tr>
</tbody>
</table>

**Figure 4. ENVR data structure**

**REQUEST=SIGNON**
Table 8. ENVROUT Storage Area Processing (continued)

<table>
<thead>
<tr>
<th>ENVR object storage area length</th>
<th>ENVR object storage area address</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nonzero</td>
<td>Nonzero</td>
<td>RACF attempts to use the storage area provided. If the area is too small to contain the ENVR object, RACF frees the storage area provided and obtains the minimum storage size needed to contain the ENVR object. Storage size is returned in the ENVR object storage area length. Storage address is returned in the ENVR object storage area address.</td>
</tr>
</tbody>
</table>

Storage is obtained and freed in the subpool and key specified in the ENVR data structure.

Since the ENVR object length is returned to the caller, the ENVR object can be moved from one storage area to another. This value is supplied as an output to the caller. RACF does not attempt to use this value in either ENVROUT or ENVRIN processing.

The caller is responsible for freeing the ENVR object storage area when it is no longer needed. The length, address, subpool, and key to be used when doing the FREEMAIN are contained in the ENVR data structure.

The ENVR object returned by ENVROUT= is a relocatable object that can be copied from one storage location to another. The returned ENVR object, or a copy of the returned ENVR object, can be specified as input to the RACROUTE interface using the ENVRIN keyword, or to the initACEE callable service using the ENVR_in parameter.

The ENVR object can be passed to other systems, but this should be done with great care. The ENVR object should not be saved for a long period of time before being used as ENVRIN, and it should not be passed to systems that have different security information. The other systems should share the RACF database and have compatible RACF installation exits and class descriptor tables.

,GROUP=address
specifies the group provided by the user who has entered the system. The address points to a 1-byte length field, followed by the group name which can be up to eight characters.

The GROUP name is part of the key which uniquely identifies which signed-on list entries are being processed.

**Note:**
1. For TYPE=SIGNIN, if GROUP is not specified, the signed-on list entry contains blanks in the GROUP field.
2. For TYPE=SIGNOFF, if GROUP is not specified, the value used to match entries in the signed-on-from list is defaulted to blanks.
3. For TYPE=SIGNOFF, the GROUP name can be specified as * to indicate a “don’t care” condition that allows a match with any GROUP name in the list. All entries matching USERID are removed from the signed-on list regardless of GROUP.

,LSTTYPE=ONFROM
specifies that you are requesting the operation for the signed-on-from list.
REQUEST=SIGNON

Although LSTTYPE has a default of ONFROM, you should specify this keyword since omitting it causes an MNOTE and an assembler return code of 4.

,POE=port-of-entry address
    specifies the address of the port of entry into the system. The address points to the partner LU name. The port of entry is an 8-character field which is left-justified and padded with blanks.

    The POE name is part of the key which uniquely identifies which signed-on list entries are being processed.

    Note: For LISTDEL and SIGNOFF, the POE name can be specified as *; this indicates a “don’t care” condition that allows a match with any POE name in the list.

,POENET=network name address
    specifies the address of a structure that consists of a 1-byte length field followed by up to an 8-byte field containing the network name of the partner LU. When specified with the POE parameter, the value specified for POENET is combined with the value specified for POE to create a network qualified name in the form netid.luname. The network qualified LU name is then used as the POE value during further processing. Note that an asterisk (*) should not be specified for POENET as it is not treated as a generic character and results in an incorrect network qualified LU name. To specify the POENET parameter, you must specify RELEASE=2.6.

,SECLABL=address
    specifies the address of an 8-byte field which contains the user's security label. The field is left-justified and padded to the right with blanks.

    Note: If SECLABL is not specified, the signed-on list entry contains blanks in the SECLABL field.

,TOKNOUT=address
    specifies the address of a user-provided area in which an output UTOKEN is to be built. The UTOKEN along with the APPL parameter can be used on a subsequent RACROUTE REQUEST=VERIFY, ENVIR=CREATE to re-create the security environment. However, using ENVROUT to obtain a copy of the security environment and then using ENVRIN to provide this security environment on a subsequent RACROUTE REQUEST=VERIFY provides better performance if the requirements for using ENVRIN are met.

,TYPE=LISTCRT
,TYPE=LISTDEL
,TYPE=SIGNIN
,TYPE=SIGNOFF
,TYPE=QSIGNON
    specifies the action to be performed for the specified APPL and POE.

    LISTCRT
        Create and initialize a signed-on-from list to represent users signed on from a specified POE to a specified APPL; use of LISTCRT is optional, as SIGNIN can perform this function automatically.

    LISTDEL
        Delete a signed-on-from list. TYPE=LISTDEL can be called while running in SRB mode.
REQUEST=SIGNON

QSIGNON
Determine if the specified user is in the signed-on-from list and optionally return data using the ACEE, ENVROUT, and TOKNOUT keywords.

SIGNIN
Add a user to the signed-on-from list; additionally, if no signed-on-from list exists for the specified APPL and POE, SIGNIN creates and initializes one.

This allows signed-on-from lists to be created implicitly (on the first TYPE=SIGNIN request), or explicitly (using TYPE=LSTCRT).

SIGNOFF
Remove users from signed-on-from lists associated with the specified APPL.

The following table indicates the various combinations you can specify to remove entries from the signed-on-from list(s). On the TYPE=SIGNOFF request, for the POE, USERID, and GROUP keywords you can do one of the following:
• Specify the keyword with an exact value.
• Specify the keyword as * ("don't care.").

In the following table, M indicates the keyword was specified with an exact value (which would be blanks for the GROUP keyword if it is not specified); * indicates that all values for that keyword are considered a match.

Table 9. Delete Actions Based on Supplied Input

<table>
<thead>
<tr>
<th>POE or POENET.POEM</th>
<th>USERID</th>
<th>GROUP</th>
<th>DELETES...</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>*</td>
<td>*</td>
<td>ALL entries</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
<td>M</td>
<td>Entries matching GROUP</td>
</tr>
<tr>
<td>*</td>
<td>M</td>
<td>*</td>
<td>Entries matching USERID</td>
</tr>
<tr>
<td>*</td>
<td>M</td>
<td>M</td>
<td>Entries matching USERID and GROUP</td>
</tr>
<tr>
<td>M</td>
<td>*</td>
<td>*</td>
<td>Entries matching POE</td>
</tr>
<tr>
<td>M</td>
<td>*</td>
<td>M</td>
<td>Entries matching POE and GROUP</td>
</tr>
<tr>
<td>M</td>
<td>M</td>
<td>*</td>
<td>Entries matching POE and USERID</td>
</tr>
<tr>
<td>M</td>
<td>M</td>
<td>M</td>
<td>Entries matching POE, USERID and GROUP</td>
</tr>
</tbody>
</table>

*,USERID=address
specifies the user identification of the user who has entered the system. The address points to a 1-byte length field, followed by the user ID which can be up to eight characters.

The value of USERID is part of the key which is used to uniquely identify which signed-on list entries are being processed.

Note: For TYPE=SIGNOFF, USERID can be specified as * to indicate a “don’t care” condition that allows a match with any USERID in the list. All entries matching GROUP are removed from the signed-on list regardless of USERID.

*,VERBEXIT=address of fullword
specifies the address of an address of a user exit routine which receives control
from RACF during SIGNOFF processing. The user exit is responsible for issuing the ALLOCATE of the SIGNOFF TP for the partner LU. When the user exit receives control from RACF, register 1 points to a parameter list mapped by ICHSGX1P, the mapping macro for the TYPE=SIGNOFF VERBEXIT. See “SGX1P: RACROUTE REQUEST=SIGNON Parameter List Mapping” in z/OS Security Server RACF Data Areas.

RACROUTE REQUEST=SIGNON, TYPE=SIGNOFF processing expects to pass control to a VERBEXIT. You can supply the VERBEXIT when the signed-on list is initially created or on a TYPE=SIGNOFF request; if specified on both, the VERBEXIT specified with TYPE=SIGNOFF is used.

If no exit address (or an address of zero) is passed, the signoff routine for the partner LU can not be invoked. The RACROUTE return code (08,10,30) indicates a failure. However, the entries are deleted from the list.

 MF=S

specifies the standard form of the RACROUTE REQUEST=SIGNON macro instruction.

Return codes and reason codes

When control is returned, space for the return code and reason code is reserved in the first two words of the RACROUTE parameter list. You can use the ICHSAFP mapping macro to access them by loading the ICHSAFP pointer with the label that you specified on the execute form of the macro. When control is returned, register 15 contains the SAF return code.

If the “don't care” character is specified in one or more keywords on a SIGNOFF and you encounter an internal error reason code, it might be possible that all user entries matching your keyword specification were not deleted. This can be verified using RACF’s DISPLAY SIGNON operator command. If you find you still have user entries to delete, you might want to rerun the request, try RACF’s SIGNOFF operator command, or try the request multiple times for specific user entries. If these still do not remove your user entries, you first need to work with the IBM support center to resolve the underlying problem.

Note: All return and reason codes are shown in hexadecimal. Also, note that SAF return code is presented as SAF RC and RACF return code is presented as RACF RC in the following section.

<table>
<thead>
<tr>
<th>SAF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACROUTE REQUEST=SIGNON has completed successfully</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>The RACROUTE REQUEST=SIGNON function was performed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Indicates a normal completion.</td>
</tr>
<tr>
<td>04</td>
<td>TYPE=SIGNOFF requested, user not found for specified port of entry.</td>
</tr>
<tr>
<td>08</td>
<td>TYPE=LISTDEL requested, list not found.</td>
</tr>
</tbody>
</table>
REQUEST=SIGNON

0C    TYPE=LISTCRT requested, list already exists.
10    TYPE=SIGNIN requested, user already signed on.
14    TYPE=LISTDEL requested, error physically deleting list(s).
       Data space storage could not be freed but RACF considers
       the list to be deleted.

04    Requested function could not be completed. No RACF decision.

RACF RC
Meaning
00    RACF could not process RACROUTE REQUEST=SIGNON request.

Reason Code
Meaning
00    RACF was not called to process the request because one of
       the following occurred:
       • The combination of class, REQSTOR, and SUBSYS was
         found in the RACF router table, and ACTION=None
         was specified.
       • The RACROUTE issuer specified DECOUPL=YES and a
         RELEASE= keyword with a higher release than is
         supported by this level of z/OS.
       • Environment is not MVS/ESA 4.2.2 or later.
       • A version of RACF earlier than 1.9.2 is installed.
       • RACF is not installed.
       • RACF is not active.

04    RACF could not complete processing of SIGNON request. If the
       reason code indicates an internal error, contact your support center.

Reason Code
Meaning
04    Unable to establish recovery or an abend occurred.
08    RACF subsystem initialization problem. One of the
       following has occurred.
       • RACF subsystem not started at IPL,
       • RACF subsystem failed to initialize correctly, or
       • RACF subsystem was abnormally stopped.
0C    Internal error: Null ID found.
10    STORAGE PROBLEM—The underlying data space storage
       management request failed.
14    STORAGE UNAVAILABLE—No more data space storage
       available for expanding the PV list.
18    Internal error: INVALID LENGTH Exception
1C    Internal error: INCONSISTENCY Exception 1.
24    Internal error: INCONSISTENCY Exception 2.
28    Internal error: OPERATION REJECTED Exception.
2C    Internal error: INVALID CONTROL DATA Exception.
34    Local lock was held by invoker.
REQUEST=SIGNON

38     Lock prematurely released, (i.e. not by obtainer).
3C     Internal error: Unexpected Exception.
40     Internal error: INVALID OFFSET Exception.
44     Internal error: INVALID KEY DEFINITION Exception.
48     Exceeded maximum number of APPL names for the signed-on lists.
08     Requested function has failed.

RACF RC

Meaning

08     User information not returned on a TYPE=QSIGNON request.

Reason Code

Meaning

04     User not found in signed-on list.
0C     The VERBEXIT returned a nonzero return code. RACF has propagated the return code (xx) back to the caller as a reason code.

Reason Code

Meaning

xx     Defined and documented by APPC/MVS VERBEXIT or other VERBEXIT being used. For possible values of xx, refer to documentation of VERBEXIT being used.

Note: List processing occurs before VERBEXIT processing. The user is deleted from the list prior to this failure.

10     Parameter list error

Reason Code

Meaning

04     No APPL LU name specified.
08     No POE LU name specified.
0C     Blank or null APPL LU name specified.
10     Blank or null POE LU name specified.
14     No TYPE was specified, or type was specified to the macro incorrectly.
18     USERID was not specified on a SIGNIN, SIGNOFF, or QSIGNON request.
1C     Blank or null USERID was specified on a SIGNIN, SIGNOFF, or QSIGNON request.
20     USERID length specified is not valid. The 1-byte length field specified was less than or equal to zero, or was greater than eight.
24     GROUP length specified is not valid. The 1-byte length field specified was less than or equal to zero, or was greater than eight.
28     The ENVR Object storage area address specified with the ENVRIN keyword is 0 (TYPE=SIGNIN), or the ENVR
REQUEST=SIGNON

Object storage area address specified is not on a doubleword boundary (TYPE=SIGNIN or TYPE=QSIGNON).

2C The length specified for the ENVR Object is greater than the length of the storage containing the ENVR Object for the ENVRIN keyword on a SIGNIN.

30 No VERBEXIT address found. The VERBEXIT must have been provided by the time the SIGNOFF is performed. Must be specified either when the list is initially created (by LISTCRT or SIGNIN), or on the SIGNOFF request.

Note: List processing occurs before VERBEXIT processing. The user is deleted from the list prior to this failure.

34 Either the POE, USERID, or GROUP keyword was incorrectly specified as * on a LISTCRT or a SIGNIN request. A "don't care" character can only be specified on a SIGNOFF or LISTDEL request.

14 RACROUTE REQUEST=VERIFY error encountered during QSIGNON.

Reason Code
Meaning

XXXXYYYY
For the explanation of this reason code, refer to RACROUTE REQUEST=VERIFY "Return codes and reason codes" on page 232. Under SAF return code of X'08', see RACF return code YYYY and RACF reason code XXXX

Example 1
Sign a user into the signed-on-from list (standard form of the macro).

RACROUTE REQUEST=SIGNON,
RELEASE=1.9.2, X
WORKA=RACWRKA, X
TYPE=SIGNIN, X
LSTTYPE=ONFROM,
APPL=MYAPPL,
POE=MYPOE,
USERID=MYUSERID,
GROUP=MYGROUP,
ENVRINIENVRDATA, X
MF=S

Example 2
List form of a RACROUTE REQUEST=SIGNON.

LISTBEG RACROUTE REQUEST=SIGNON,
RELEASE=1.9.2, X
WORKA=RACWRKA, X
TYPE=SIGNIN, X
LSTTYPE=ONFROM,
APPL=MYAPPL,
POE=MYPOE,
USERID=MYUSERID,
GROUP=MYGROUP,
ENVRIN=ENVRDATA, X
MF=L
Example 3
Use QSIGNON to determine if a user is signed in, and if so, obtain an ENVROUT object (execute form of the macro).

```
  RACROUTE REQUEST=SIGNON, X
  RELEASE=1.9.2, X
  WORKA=RACRWRKA, X
  TYPE=QSIGNON, X
  LSTTYPE=ONFROM, X
  ENVROUT=ENVRDATA, X
  MF=(E,LISTBEG)
```

DATA DECLARATIONS

```
  DS 0D
  RACRWRKA DC 128F'0' 512 BYTE RACROUTE WORK AREA
  MYUSERID DC AL1(7) LENGTH OF USERID
  DC CL7'DANHERE' THE ACTUAL USERID
  MYGROUP DC AL1(6) LENGTH OF GROUP NAME
  DC CL6'DEPT52' THE ACTUAL GROUP NAME
  MYSECLABL DC CL8'SECRET' SECURITY LABEL
  MYTOKNOUT DS 10D 80 BYTES FOR RETURNED TOKEN
  MYVERBEXIT DC A(0) ADDRESS OF VERBEXIT
  MYPOE DC CL8'DANIWS' PARTNER LU NAME
  MYAPPL DC CL8'MVSHOST1' LOCAL LU NAME
  MYACEE DC A(0) ADDRESS OF RETURNED ACEE
  ENVRDATA DS 0F
  DC F'0' ENVR LENGTH
  DC F'4096' DESIRED ENVR STORAGE AREA SIZE
  DC A(0) LET RACF OBTAIN IT
  DC AL1(1) IN SUBPOOL 1
  DC AL1(1) IN KEY 1
```

RACROUTE REQUEST=SIGNON (list form)

The list form of the RACROUTE REQUEST=SIGNON macro instruction is written as follows. Refer to the standard form of the RACROUTE REQUEST=SIGNON macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

```
  REQUEST=SIGNON
  ,TYPE=LISTCRT
  ,TYPE=LISTDEL
  ,TYPE=SIGNIN
  ,TYPE=SIGNOFF
  ,TYPE=QSIGNON
  ,POE=port of entry addr
  ,POENET=network name addr
  ,APPL='applname'
```

name symbol. Begin name in column 1.

b One or more blanks must precede RACROUTE

RACROUTE

b One or more blanks must follow RACROUTE
REQUEST=SIGNON

,APPL=applname addr  applname addr: A-type address
Note: TYPE, POE and APPL are required on or before RACROUTE REQUEST=SIGNON is issued with MF=E.

,LSTTYPE=ONFROM  Default: LSTTYPE=ONFROM

,LF=.

If TYPE=LISTCRT is specified:

,VERBEXIT=address of fullword  address of fullword: A-type address

If TYPE=SIGNIN is specified:

,USERID=userid addr  userid addr: A-type address

,ENVRIN=envr data addr  envr data addr: A-type address

,GROUP=group addr  group addr: A-type address

,SECLABL=seclabel addr  seclabel addr: A-type address

,VERBEXIT=address of fullword  address of fullword: A-type address

If TYPE=SIGNOFF is specified:

,USERID=userid addr  userid addr: A-type address

,GROUP=group addr  group addr: A-type address

,SECLABL=seclabel addr  seclabel addr: A-type address

,VERBEXIT=address of fullword  address of fullword: A-type address

If TYPE=QSIGNON is specified:

,USERID=userid addr  userid addr: A-type address

,GROUP=group addr  group addr: A-type address

,SECLABL=seclabel addr  seclabel addr: A-type address

,TOKNOUT=utoken addr  utoken addr: A-type address

,ACEE=address of fullword  address of fullword: A-type address

,ENVROUT=envr data addr  envr data addr: A-type address

The parameters are explained under the standard form of the SIGNON macro instruction with the following exception:
REQUEST=SIGNON

\texttt{,MF =L}

specifies the list form of the SIGNON macro instruction. The “L” form defines an area to be used for the parameter list.

**RACROUTE REQUEST=SIGNON (execute form)**

The execute form of the RACROUTE REQUEST=SIGNON macro instruction is written as follows. Refer to the standard form of the RACROUTE REQUEST=SIGNON macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

\begin{verbatim}
  name
  name symbol. Begin name in column 1.

  b
  One or more blanks must precede RACROUTE

  RACROUTE
  b
  One or more blanks must follow RACROUTE

  REQUEST=SIGNON
  ,TYPE=LISTCRT
  ,TYPE=LISTDEL
  ,TYPE=SIGNIN
  ,TYPE=SIGNOFF
  ,TYPE=QSIGNON
  ,POE=
    port of entry addr
  ,POENET=
    network name addr
  ,APPL=
    applname
  ,APPL=
    applname addr
  ,LSTTYPE=ONFROM
  ,MF=(E,ctrl addr)

  If \texttt{TYPE=LISTCRT} is specified:

  ,VERBEXIT=
    address of fullword

  If \texttt{TYPE=SIGNIN} is specified:

  ,USERID=
    userid addr
  ,ENVRIN=
    envr data addr
  ,GROUP=
    group addr
  ,SECLABL=
    seclabel addr
  ,VERBEXIT=
    address of fullword
\end{verbatim}

\texttt{Note: \texttt{TYPE}, \texttt{POE} and \texttt{APPL} are required on or before RACROUTE REQUEST=SIGNON is issued with \texttt{MF=E}.}
If TYPE=SIGNOFF is specified:

- USERID=userid addr
  - userid addr: RX-type address or register (2) – (12)
- GROUP=group addr
  - group addr: RX-type address or register (2) – (12)
- SECLABL=SECLABL addr
  - seclabl addr: RX-type address or register (2) – (12)
- VERBEXIT=address of fullword
  - address of fullword: RX-type address or register (2) – (12)

If TYPE=QSIGNON is specified:

- USERID=userid addr
  - userid addr: RX-type address or register (2) – (12)
- GROUP=group addr
  - group addr: RX-type address or register (2) – (12)
- SECLABL=seclabel addr
  - seclabel addr: RX-type address or register (2) – (12)
- TOKNOUT=utoken addr
  - utoken addr: RX-type address or register (2) – (12)
- ACEE=address of fullword
  - address of fullword: RX-type address or register (2) – (12)
- ENVROUT=envr data addr
  - envr data addr: RX-type address or register (2) – (12)

The parameters are explained under the standard form of the SIGNON macro instruction with the following exception:

- MF=(E,ctrl addr)
  - specifies the execute form of the SIGNON macro where ctrl addr is the address of the associated parameter list.

RACROUTE REQUEST=SIGNON (modify form)

The modify form of the RACROUTE REQUEST=SIGNON macro instruction is written as follows. Refer to the standard form of the RACROUTE REQUEST=SIGNON macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE</td>
</tr>
<tr>
<td>RACROUTE</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>One or more blanks must follow RACROUTE</td>
</tr>
<tr>
<td>REQUEST=SIGNON</td>
<td></td>
</tr>
</tbody>
</table>
REQUEST=SIGNON

.,TYPE=LISTCRT
.,TYPE=LISTDEL
.,TYPE=SIGNIN
.,TYPE=SIGNOFF
.,TYPE=QSIGNON

.,POE=port of entry addr  port of entry addr: RX-type address or register (2) – (12)
.,POENET=network name addr
.,APPL=applname' applname: 1–8 character name
.,APPL=applname addr

Note: TYPE, POE and APPL are required on or before RACROUTE REQUEST=SIGNON is issued with MF=E.

.,LSTTYPE=ONFROM

.,MF=(M,ctrl addr)

If TYPE=LISTCRT is specified:

.,VERBEXIT=address of fullword

If TYPE=SIGNIN is specified:

.,USERID=userid addr
.,ENVRIN=envr data addr
.,GROUP=group addr
.,SECLABL=seclabel addr
.,VERBEXIT=address of fullword

If TYPE=SIGNOFF is specified:

.,USERID=userid addr
.,GROUP=group addr
.,SECLABL=seclabel addr
.,VERBEXIT=address of fullword

If TYPE=QSIGNON is specified:

.,USERID=userid addr
.,GROUP=group addr
.,SECLABL=seclabel addr
.,TOKNOUT=utoken addr
.,ACEE=address of fullword
REQUEST=SIGNON

,ENVROUT=envr data
  envr data addr: RX-type address or register (2) – (12)

The parameters are explained under the standard form of the SIGNON macro instruction with the following exception:

,\MF=(M,\ctrl addr)
  specifies the modify form of the SIGNON macro. The “M” form generates code to put the parameters into the parameter list specified by \ctrl addr, the address of the associated parameter list.

RACROUTE REQUEST=STAT: Determine RACF Status

The RACROUTE REQUEST=STAT macro determines if RACF is active and, optionally, determines whether a given resource class is defined to RACF. If a resource class name is defined to RACF, the macro also determines whether the class is active.

RACROUTE REQUEST=STAT is a branch-entered service that uses standard linkage conventions.

To use this service, you must also specify RELEASE=1.9 or a later release number.

RACROUTE REQUEST=STAT (standard form)

The standard form of the RACROUTE REQUEST=STAT macro is written as follows. For a description of additional keywords that you can code and additional parameters that are required on the RACROUTE request, but that are not specific to this request type, see “RACROUTE (standard form)” on page 13.

| \name | name: Symbol. Begin \name in column 1.
| b | One or more blanks must precede RACROUTE.

RACROUTE

| b | One or more blanks must follow RACROUTE.

REQUEST=STAT

,\CLASS=class name’
  class name: 1–8 character class name

,\CLASS=class name addr
  class name addr: A-type address or register (2) – (12)

,\COPY=copy addr
  copy addr: A-type address or register (2) – (12)

,\COPYLEN=copy length
  copy length: length or register (2) – (12)

,\ENTRY=entry addr
  entry addr: A-type address or register (2) – (12)

,\NEXT=next class addr
  next class addr: A-type address or register (2) – (12)

,\MF=S

The parameters are explained as follows:
\texttt{\texttt{CLASS='class name'}}, \texttt{CLASS=class name addr}

specifies the class name to be located. The name can be explicitly defined on the macro by enclosing the name in quotes. If specified, the address must point to an 8-byte field containing the class name, left-justified and padded with blanks if necessary.

The class name specified must be a general resource defined to RACF in the class descriptor table (CDT). (For a list of the classes supplied by IBM, see Appendix E, “Supplied class descriptor table entries,” on page 397.)

\textbf{Note:}

1. The classes DATASET, USER, and GROUP are not in the class descriptor table.

2. If you specify both the \texttt{CLASS} and \texttt{NEXT} parameters, the \texttt{NEXT} parameter will be ignored.

3. If \texttt{CLASS=} and \texttt{NEXT=} are omitted, the status of RACF is returned.

\texttt{COPY=}

specifies the address, or register containing the address, of storage obtained by the caller. It is the responsibility of the caller to obtain this storage before the \texttt{RACROUTE REQUEST=STAT} call and to free it when it is no longer needed.

On return from the macro request, the storage area contains a copy of the class entry from the class descriptor table mapped by ICHPCNST.

\texttt{COPYLEN=}

specifies the length, or a register containing the length, of the \texttt{COPY} area requested.

For applications requesting a complete class entry, the length of the \texttt{COPY} area provided by the caller needs to change whenever the \texttt{CNST} or \texttt{CNSX} lengths change. When that occurs, callers are not required to recompile their programs if they use field \texttt{RCVTCDDL} which contains the length needed to hold both a \texttt{CNST} and a \texttt{CNSX} entry.

If the value for \texttt{COPYLEN} is smaller than field \texttt{RCVTCDDL}, then the \texttt{CNST} and \texttt{CNSX} data returned is truncated to the length specified on the \texttt{COPYLEN} operand.

If the value for \texttt{COPYLEN} is larger than field \texttt{RCVTCDDL}, the \texttt{CNST} and \texttt{CNSX} class entries are returned. The rest of the \texttt{COPY} area is unaltered.

If the value of \texttt{COPYLEN} is 0, no data is returned and the \texttt{COPY} area is unaltered.

\textbf{Note:} The \texttt{COPY} and \texttt{COPYLEN} operands allow AMODE 24 callers to access the fields located above the 16 megabyte line contained in the \texttt{CNSX}.

\textbf{Restrictions:}

1. The \texttt{COPY} and \texttt{COPYLEN} operands are ignored when the \texttt{CLASS=} and \texttt{NEXT=} operands are omitted.

2. The \texttt{COPY} and \texttt{COPYLEN} operands require that \texttt{RELEASE=2.2} or later be specified on the \texttt{RACROUTE REQUEST=STAT} macro.

3. You must specify both the \texttt{COPY=} and \texttt{COPYLEN=} operands. If only one of the operands is specified, the one that is specified is ignored.
REQUEST=STAT

<table>
<thead>
<tr>
<th>ENTRY=entry addr</th>
</tr>
</thead>
<tbody>
<tr>
<td>specifies the address of a 4-byte area that is set to the address of the specified class in the static class descriptor table. If the class specified in the CLASS= operand is found in the dynamic class descriptor table, zero will be returned for entry addr.</td>
</tr>
</tbody>
</table>

**Rules:**

1. AMODE 24 callers should use the COPY and COPYLEN operands instead of the ENTRY operand.
2. Use the COPY= and COPYLEN= operands for classes in the dynamic class descriptor table.
3. If you specify ENTRY=, it will be ignored when the CLASS= operand is omitted.
4. If you specify both ENTRY= and NEXT=, the ENTRY= operand is ignored.

<table>
<thead>
<tr>
<th>NEXT=next class addr</th>
</tr>
</thead>
<tbody>
<tr>
<td>specifies that RACF is to retrieve the class descriptor entry for the class following (in alphabetical order) the class specified. The address must point to an 8-byte field containing the class name, left-justified and padded with blanks, if necessary. If completed successfully, the NEXT operation updates the area pointed to by the NEXT operand with the name of the class just processed. The NEXT= function will consider the dynamic class descriptor table, if it exists, as a logical extension to the static class descriptor table. Entries in both tables will be considered while locating the next alphabetical entry. Alphabetical order is determined by the EBCDIC ordering of the characters in the class name, as illustrated in Table 10 on page 181.</td>
</tr>
</tbody>
</table>
Table 10. Alphabetic sort order for NEXT= processing

<table>
<thead>
<tr>
<th>Sorting order of characters allowed in a class name</th>
<th>EBCDIC equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>X'5B'</td>
</tr>
<tr>
<td>#</td>
<td>X'7B'</td>
</tr>
<tr>
<td>@</td>
<td>X'7C'</td>
</tr>
<tr>
<td>uppercase letters A–Z</td>
<td>X'C1'–X'E9'</td>
</tr>
<tr>
<td>numbers 0–9</td>
<td>X'F0'–X'F9'</td>
</tr>
</tbody>
</table>

When you specify NEXT=, RACF will locate the class that alphabetically follows the specified class in the class descriptor table. RACF will return the class descriptor entry of that 'next' class in the areas specified on the COPY= and COPYLEN= operands. If you do not specify COPY= and COPYLEN=, the return and reason codes will still reflect the status of RACF and the class returned in the NEXT= operand.

To retrieve the first class in the class descriptor table, the NEXT field must be initialized to either eight blanks (X'40') or eight null characters (X'00'). If a class name is specified, it will be used as a search string for comparison to existing class names. Regardless of whether RACF does or does not find the specified class name in the table, RACF will return the next name alphabetically greater than the specified name, if one exists in the table.

**Note:** The classes DATASET, USER, and GROUP are not in the class descriptor table, so they will not be returned by the NEXT= function.

**Rules:**
1. If you specify NEXT=, do not specify CLASS= nor ENTRY=.
   a. NEXT= will be ignored if it is specified with CLASS=.
   b. ENTRY= will be ignored if it is specified with NEXT=.
2. When specifying NEXT=, you must also specify RELEASE=7709 or later.

,MF=S

specifies the standard form of the RACROUTE REQUEST=STAT macro instruction.

**Return codes and reason codes**

When you execute the macro, space for the RACF return code and reason code is reserved in the first two words of the RACROUTE parameter list. You can access them using the ICHSAFP mapping macro, by loading the ICHSAFP pointer with the label that you specified on the list form of the macro. When control is returned, register 15 contains the SAF return code.

**Note:** All return and reason codes are shown in hexadecimal. Also, note that SAF return code is presented as SAF RC and RACF return code is presented as RACF RC in the following section.

<table>
<thead>
<tr>
<th>SAF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACROUTE REQUEST=STAT has completed successfully.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
REQUEST=STAT

00  RACF is active and the class (specified in CLASS= or returned in NEXT=) is active.

04  RACROUTE REQUEST=STAT did not complete successfully.

RACF RC

Meaning

00  No security decision could be made.

Reason Code

Meaning

00  RACF was not called to process the request because one of the following occurred:
  • RACF is not installed.
  • The combination of class, REQSTOR, and SUBSYS was found in the RACF router table, and ACTION=NONE was specified.
  • The RACROUTE issuer specified DECOUPL=YES and a RELEASE= keyword with a higher release than is supported by this level of z/OS.

04  RACF is active; the class (specified in CLASS= or returned in NEXT=) is inactive.

08  RACF is active. If CLASS= was specified, the class is not defined to RACF. If NEXT= was specified, no class was found to satisfy the request; the end of the class descriptor table was found.

0C  RACF is inactive; the class (specified in CLASS= or returned in NEXT=) was active prior to RACF being deactivated.

10  RACF is inactive; the class (specified in CLASS= or returned in NEXT=) is inactive.

14  RACF is inactive. If CLASS= was specified, the class is not defined to RACF. If NEXT= was specified, no class was found to satisfy the request; the end of the class descriptor table was found.

18  RACF is not installed, or an insufficient level of RACF is installed.

1C  Incorrect parameter-list length is detected for the request-specific portion of the RACROUTE REQUEST=STAT parameter list.

64  Indicates that the CHECK subparameter of the RELEASE keyword was specified on the execute form of the RACROUTE REQUEST=STAT macro; however, the list form of the macro does not have the same RELEASE parameter. Macro processing terminates.

Note: For RACF return codes X'00', X'04', X'0C', and X'10', the class-descriptor entry for the specified class is returned to the caller in the COPY area provided or in the area pointed to by the ENTRY address.

Example 1

Determine whether the DASDVOL class is active, and retrieve the address of its class descriptor. A fullword, CDADDR, contains the class-descriptor address.

RACROUTE REQUEST=STAT,CLASS='DASDVOL',ENTRY=CDADDR, X
  WORKA=RACWK

: RACWK DS CL512
Example 2
Retrieve a copy of the CNST and CNSX area and reference fields in the CNSX.

```plaintext
USING CVTMAP,4 * Establish addressability to CVT
L 4,16(0)
USING RCVT,5 * Establish addressability to RCVT
L 5,CVTRAC
L 2,RCVTCDTL * Load length to GETMAIN in reg 2.
GETMAIN RU,LV=(2) * Get storage.
LR 3,1 * Save address of new area in reg 3.
RACROUTE REQUEST=STAT,WORKA=SAFWORK,
   CLASS='TCICSTRN',
   COPY=(3), X
   COPYLEN=(2), X
   RELEASE=2.2,MF=S
USING CNST,3 * Reg 3 was set up to point to COPY area
USING CNSX,4 * Establish addressability to CNSX
L 4,CNSTCNSX
IC 8,CNSTDFRC * Load value of CNSTDFRC
ICHPCNST * Generates CNST and CNSX DSECTs
```

Example 3
Use the COPY and ENTRY operands together in a program that runs on RACF Version 2 Release 2 (FMID HRF2220) and on releases prior to RACF Version 2 Release 2.

```plaintext
LA 0,COPYAREA * Get address of COPY area
LA 1,AREALEN * Get length of COPY area
SLR 15,15 * Clear length/padding register
MVCL 0,14 * Zero out COPY area
RACROUTE REQUEST=STAT,WORKA=SAFWORK,
   CLASS='JESSPOOL',
   ENTRY=EPTR,
   COPY=COPYAREA, COPYLEN=AREALEN,
   RELEASE=2.2, MF=S
***
*** Check if class data was returned in COPYAREA.
*** -- If so, use COPYAREA data mapped by ICHPCNST macro.
*** -- Otherwise, use class entry returned in EPTR because
*** this system is running an earlier release of RACF that
*** does not support COPY function.
***
LA 3,COPYAREA * Get address of COPY area
USING CNST,3
CLC CNSTLGT,ZEROH * Was class data returned in
   COPYAREA?
BNE USECOPY * -- Yes,
   USE COPYENTRY
   USECNST,3 * Use class entry in COPYAREA
   USE CNSTDFRC,4 * -- No,
   USE entry pointer EPTR
   USE CNSTCNSX,4
L 3,EPTR * and pre-2.2 CNST mapping

USECOPY EQU * * Use class entry in COPYAREA
USING CNST,3
USING CNSX,4 * Establish addressability to CNSX
L 4,CNSTCNSX * if program needs to access CNSX
   ** fields

COPYAREA DS CL400 * Large enough to hold a 2.2 class entry
AREALEN EQU 400 * Size of area provided (COPYAREA)
EPTR DS A * Address of class ENTRY returned
ZEROH DC H'0'
OLDCNST DSECT * Pre-2.2 CNST declared locally
```
Example 4
Use the NEXT operand to process all entries, both static and dynamic, in the class descriptor table.

* MVC CLASSNM,BLANKS Set classname to blanks
LA 4,COPYAREA
USING CNST,4 Addressability to CNST fields
LA 5,COPYCNSX Addressability to CNSX fields
* LA 0,COPYAREA
LA 1,COPYLEN
SLR 15,15 Clear the return area for CDT entry
* MVCL 0,14
* LOOP EQU *
* Issue RACROUTE REQUEST=STAT to get next class.
* CLSSTAT RACROUTE REQUEST=STAT,COPY=COPYAREA,COPYLEN=COPYLEN,
 X NEXT=CLASSNM,WORKA=RRWA,
 X RELEASE=7709,MF=S
* L 6,CNSTLGT See if we have reached the end
LTR 6,6
BZ ALLDONE
* Process each class here. A copy of the CDT entry is in COPYAREA,
* and the class name is also in CLASSNM.
* B LOOP
ALLDONE EQU * End of CDT entries
:
CLASSNM DS CL8 Name of class from REQUEST=STAT
BLANKS DC CL8' ' Name of class from REQUEST=STAT
COPYAREA DS CL(CNSTCBLN) Area for CNST portion
COPYCNSX DS CL(CNSXCBLN) Area for CNSX portion
COPYLEN EQU =-COPYAREA Length of CDT entry
RRWA DS CLS12 RACROUTE work area
ICHPCNST Generates CNST and CNSX DSECT

RACROUTE REQUEST=STAT (list form)
The list form of the RACROUTE REQUEST=STAT macro instruction is written as follows. Refer to the standard form of the RACROUTE REQUEST=STAT macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
</tbody>
</table>

RACROUTE

| b    | One or more blanks must follow RACROUTE. |

REQUEST=STAT

| ,CLASS='class name' | class name: 1–8 character class name |
| ,CLASS=class name addr | class name addr: A-type address |
REQUEST=STAT

,.COPY=copy addr
    copy addr: A-type address or register (2) – (12)
,.COPYLEN=copy length
    copy length: length or register (2) – (12)
,.ENTRY=entry addr
    entry addr: A-type address
,.NEXT=next class addr
    next class addr: A-type address

,MF=L

The parameters are explained under the standard form of the RACROUTE REQUEST=STAT macro with the following exception:

,.MF=L

specifies the list form of the RACROUTE REQUEST=STAT macro.

RACROUTE REQUEST=STAT (execute form)

The execute form of the RACROUTE REQUEST=STAT macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=STAT macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

name
name: Symbol. Begin name in column 1.

b
One or more blanks must precede RACROUTE.

RACROUTE

b
One or more blanks must follow RACROUTE.

REQUEST=STAT

,.RELEASE=number
    number: See standard form
,.RELEASE=(number, CHECK)
,.CLASS=class name addr
    class name addr: RX-type address or register (2) – (12)
,.COPY=copy addr
    copy addr: A-type address or register (2) – (12)
,.COPYLEN=copy length
    copy length: length or register (2) – (12)
,.ENTRY=entry addr
    entry addr: RX-type address or register (2) – (12)
,.NEXT=next class addr
    next class addr: RX-type address or register (2) – (12)
,MF=(E,ctrl addr)
    ctrl addr: RX-type address or register (1) – (12)

The parameters are explained under the standard form of the RACROUTE REQUEST=STAT macro with the following exceptions:

,.RELEASE=number
,.RELEASE=(number, CHECK)

specifies the RACF release level of the parameter list to be generated by this macro.

Chapter 3. System macros  185
REQUEST=STAT

To use the parameters associated with a release, you must specify the number of that release or a later release number. If you specify a parameter with an incompatible release level, the parameter is not accepted by the macro processing. An error message is issued at assembly time.

When you specify the RELEASE keyword, checking is done at assembly time. Compatibility between the list and execute forms of the RACROUTE REQUEST=STAT macro are validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.

,MF=(E,ctrl addr)
specifies the execute form of the RACROUTE REQUEST=STAT macro, using a remote, control-program parameter list.

RACROUTE REQUEST=STAT (modify form)

The modify form of the RACROUTE REQUEST=STAT macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=STAT macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
</tbody>
</table>

RACROUTE

| b    | One or more blanks must follow RACROUTE. |

REQUEST=STAT

,CLASS=class name addr
class name addr: RX-type address or register (2) – (12)

,COPY=copy addr
copy addr: A-type address or register (2) – (12)

,COPYLEN=copy length
copy length: length or register (2) – (12)

,ENTRY=entry addr
entry addr: RX-type address or register (2) – (12)

,NEXT=next class addr
next class addr: RX-type address or register (2) – (12)

,MF=(M,ctrl addr)
ctrl addr: RX-type address or register (1) – (12)

The parameters are explained under the standard form of the RACROUTE REQUEST=STAT macro, with the following exceptions:

,MF=(M,ctrl addr)
specifies the modify form of the RACROUTE REQUEST=STAT macro, using a remote, control-program parameter list.
RACROUTE REQUEST=TOKENBLD: Build a UTOKEN

The RACROUTE REQUEST=TOKENBLD macro builds a UTOKEN. The TOKNIN keyword specifies the location of the existing token from which a modified token is to be built. Note that the modification does not change the input token; instead, the function builds a new, modified token from the parameters provided. The TOKNIN keyword is optional; if you want to build a new UTOKEN, the TOKNIN keyword should not be used. The TOKNOUT keyword specifies the location where the new, modified token is to be built.

The following order of priority exists for building the UTOKEN:
- Keywords specified on the request take precedence over corresponding fields in the TOKNIN and STOKEN parameters.
- All fields within the token specified by the TOKNIN keyword take precedence over those specified by STOKEN.
- The fields for the submitter’s ID, submitter’s group, submit node, execution node, session, port of entry and its class, as obtained from the token specified by the STOKEN keyword, are last.

If you do not want certain fields overridden, do not specify keywords for those fields.

To use this service, you must specify RELEASE=1.9 or a later release number.

To issue the RACROUTE REQUEST=TOKENBLD macro, the calling module must be authorized (APF-authorized, in system key 0-7, or in supervisor state).

The caller cannot hold locks when issuing RACROUTE REQUEST=TOKENBLD.

This function is performed by SAF.

RACROUTE REQUEST=TOKENBLD (standard form)

The standard form of the RACROUTE REQUEST=TOKENBLD macro is written as follows. For a description of additional keywords that you can code and additional parameters that are required on the RACROUTE request, but that are not specific to this request type, see "RACROUTE (standard form)" on page 13.

```
<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

One or more blanks must precede RACROUTE.

RACROUTE

One or more blanks must follow RACROUTE.

REQUEST=TOKENBLD

,TOKNOUT=output token addr  output token addr: A-type address or register (2) – (12)

,EXENODE=execution node addr  execution node addr: A-type address or register (2) – (12)

,GROUP=group addr  group addr: A-type address or register (2) - (12)
```
REQUEST=TOKENBLD

,POE=port of entry addr
    port of entry addr: A-type address or register (2) – (12)
,POENET=network name addr
    network name addr: A-type address or register (2) – (12)
,REMOTE=YES
    Default: REMOTE=NO
,REMOTE=NO
,SECLABL=seclabel addr
    seclabel addr: A-type address or register (2) – (12)
,SESSION=type
    type: Any valid session type
    Default: SESSION=TSO
,SGROUP=submitting group addr
    submitting group addr: A-type address or register (2) – (12)
,SNODE=submitting node addr
    submitting node addr: A-type address or register (2) – (12)
,STOKEN=token addr
    token addr: A-type address or register (2) - (12)
,SUSERID=submitting userid addr
    submitting userid addr: A-type address or register (2) – (12)
,TERMID=terminal addr
    terminal addr: A-type address or register (2) – (12)
,TOKNIN=input token addr
    input token addr: A-type address or register (2) – (12)
,TRUSTED=YES
    Default: TRUSTED=NO
,TRUSTED=NO
,USERID=userid addr
    userid addr: A-type address or register (2) - (12)
,MF=S

The parameters are explained as follows:

,EXENODE=execution node addr
    specifies the address of an area that contains a 1-byte length field followed by
    the name of the node on which the unit of work is to be executed. The node
    name cannot exceed eight bytes.

,GROUP=group addr
    specifies the group of the user who has entered the system. The address points
    to a 1-byte length field followed by the group name. The group name cannot
    exceed eight bytes.

,POE=port of entry addr
    specifies the address of the port of entry into the system. The address points to
    the name of the input device through which the user or job entered the system.
    For example, this can be the name of the input device through which the job
    was submitted or the terminal logged on. The port of entry is an 8-character
    field, left-justified and padded with blanks.

    The port of entry (POE) becomes a part of the user's security token (UTOKEN).
    A flag in the UTOKEN uniquely identifies the RACF general-resource class to
    which the data in the POE field belongs: APPCPORT, TERMINAL, CONSOLE,
    or JESINPUT.
REQUEST=TOKENBLD

The RACF class JESINPUT provides the conditional access support for jobs entered into the system through a JES input device, and the CONSOLE class performs the same task for commands that originate from a console. In addition, the APPCPORT class provides conditional access support for users entering the system from a given LU (APPC port of entry).

The TERMINAL class covers the terminal used to log on to TSO.

When both the POE and TERMID keywords are specified, the POE keyword takes precedence.

-,POENET=network name address
specifies the address of a structure that consists of a 1-byte length field followed by up to an 8-byte field containing the network name of the partner LU. When specified with the POE parameter, the value specified for POENET is combined with the value specified for POE to create a network qualified name in the form netid.luname. The network qualified LU name is then used as the POE value during further processing. POENET is only valid with SESSION=APPCTP, and should not be specified with any other type of session. To specify the POENET parameter, you must specify RELEASE=2.6.

-,REMOTE=YES
-,REMOTE=NO
specifies whether the job came through the network. The default is REMOTE=NO.

-,SECLABL=seclabel addr
specifies the address of an 8-byte, left-justified field, which contains the security label, padded to the right with blanks.

To use this keyword, you must also specify RELEASE=1.9 or a later release number.

An installation can use security labels to establish an association between a specific RACF security level (SECLLEVEL) and a set of (zero or more) RACF security categories (CATEGORY). If it is necessary to use security labels to prevent the unauthorized movement of data from one level to another when multiple levels of data are in use on the system at the same time, see z/OS Security Server RACF Security Administrator's Guide for further information.

-,SESSION=type
specifies the session type to be associated with the request. Session types are literals. When the SESSION keyword is used in combination with the POE keyword, SESSION determines the class with which the POE keyword is connected.

The default session type is TSO.

The allowable session types and their associated POE classes are:

<table>
<thead>
<tr>
<th>Session type</th>
<th>Description</th>
<th>POE class</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPCTP</td>
<td>An APPC transaction program</td>
<td>APPCPORT</td>
</tr>
<tr>
<td>COMMAND</td>
<td>A command</td>
<td>CONSOLE</td>
</tr>
<tr>
<td>CONSOPER</td>
<td>A console operator</td>
<td>CONSOLE</td>
</tr>
<tr>
<td>EXTBATCH</td>
<td>A job from external reader (EXT)</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>EXTXBM</td>
<td>An execution batch monitor job</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>INTBATCH</td>
<td>A batch job from internal reader (INT)</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>INTXBM</td>
<td>An execution batch monitor job from internal reader (INT)</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>MOUNT</td>
<td>A mount</td>
<td>None</td>
</tr>
</tbody>
</table>
### SESSION TYPE DESCRIPTION

<table>
<thead>
<tr>
<th>Session type</th>
<th>Description</th>
<th>POE class</th>
</tr>
</thead>
<tbody>
<tr>
<td>NJEBATCH</td>
<td>A job from network job entry (NJE)</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>NJEOPER</td>
<td>A network job entry operator</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>NJEXBM</td>
<td>An network execution batch monitor job</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>NJSYSOUT</td>
<td>A network SYSOUT</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>OMVSSRV</td>
<td>An OMVS server application</td>
<td>None</td>
</tr>
<tr>
<td>RJEBATCH</td>
<td>A batch job from remote job entry (RJE)</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>RJEOPER</td>
<td>A remote job-entry operator</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>RJEXBM</td>
<td>A remote execution batch monitor job</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>STARTED</td>
<td>A started procedure of started task</td>
<td>None</td>
</tr>
<tr>
<td>SYSAS</td>
<td>A system address space</td>
<td>None</td>
</tr>
<tr>
<td>TKNUNKWN</td>
<td>An unknown user from NJE</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>TSO</td>
<td>A TSO or other interactive session logon</td>
<td>TERMINAL</td>
</tr>
</tbody>
</table>

**Note:** When no POE class is associated with the session type, the POE ID and session are preserved.

`:SGROUP` specifies the address of an area that contains a 1-byte length field followed by the group name of the user who submitted the unit of work. The group ID cannot exceed eight bytes.

`:SUSERID` specifies the address of an area that contains a 1-byte length field followed by the user ID of the user who submitted the unit of work. The user ID cannot exceed eight bytes.

`:STOKEN` specifies the address of the submitter's UTOKEN. The first byte contains the length of the UTOKEN, and the second byte contains the version number. See the ICHRUTKN mapping, “RUTKN: Resource/Security Token” in [z/OS Security Server RACF Data Areas](https://www.ibm.com/support/knowledgecenter/SSS7N7_2.1.0/com.ibm.zos.security.racf_210.doc/racf_data_areas.html) for the current version and release.

If you specify STOKEN, the user ID in STOKEN becomes the submitter's ID in TOKNOUT, unless you specify the submitter's ID (SUSER) keyword. In this case, that keyword becomes the submitter's ID in TOKNOUT. Likewise, if you specified GROUP in STOKEN, that becomes the submitter's group in TOKNOUT, unless you specified the submitter's group (SGROUP) keyword. The SESSION, port of entry (POE), and port-of-entry class (POEX) fields are also used from the STOKEN. The execution node becomes the resulting submit node and execution node, unless you specify the submit node (SNODE) or execution node (EXENODE) keywords. In all cases, the specified keywords on the request override the fields of STOKEN, if one is specified.

`:SNODE` specifies the address of an area that contains a 1-byte length field followed by the name of the node from which the unit of work was submitted. The node name cannot exceed eight bytes.

When OMVSSRV is specified, user profile statistics are updated daily at most. Audit records are only created when one of the following conditions are met:

- An incorrect password or password phrase is specified.
- The user ID has been revoked.
- A new password or password phrase was provided.
- A security label error occurred.
REQUEST=TOKENBLD

,TERMD=terminal_addr
specifies the address of the identifier of the terminal through which the user is accessing the system. The address points to an 8-byte area containing the terminal identifier. The area must reside in a non-task-related storage subpool.

,TOKNIN=input_token_addr
specifies the address of the UTOKEN or RTOKEN that is to be used as a base for the output token.

,TOKNOUT=output_token_addr
specifies the address of the caller-provided area for the modified token data. The first byte of storage at the address specified must contain the token length. The second byte must contain the format version of the token. This provides for downward compatibility with all versions of the token map.

For a description of the fields that are used from STOKEN by TOKNOUT, see the STOKEN description.

,TRUSTED=YES,TRUSTED=NO
specifies whether or not the unit of work is a member of the trusted computer base. Subsequent RACROUTE REQUEST=AUTH requests using a token with this attribute have the following effects:

• Authorization checking is bypassed (this includes bypassing the checks for security classification on users and data).
• No statistics are updated.
• No audit records are generated, except those requested using the SETROPTS LOGOPTIONS command or the UAUDIT operand on the ALTUSER command.
• No exits are called.

Subsequent RACROUTE REQUEST=FASTAUTH requests using a token with this attribute have the following effects:

• Authorization checking is bypassed (this includes bypassing the checks for security classification on users and data).
• No statistics are updated.
• No audit records are generated, except those requested using the UAUDIT operand on the ALTUSER command.

This is similar to having the started-procedures-table trusted bit on.

Note: The TRUSTED=YES keyword only has meaning when SESSION=STARTED is also specified.

,USERID=userid_addr
specifies the identification of the operator who has entered the system. The address points to a 1-byte length field followed by the user ID. The user ID cannot exceed eight bytes.

,MF=S
specifies the standard form of the RACROUTE REQUEST=TOKENBLD macro instruction.

Return codes and reason codes
When you execute the macro, space for the RACF return code and reason code is reserved in the first two words of the RACROUTE parameter list. You can access them using the ICHSAFP mapping macro, by loading the ICHSAFP pointer with the label that you specified on the list form of the macro. When control is returned,
REQUEST=TOKENBLD

register 15 contains the SAF return code.

| Note: All return and reason codes are shown in hexadecimal. Also, note that SAF return code is presented as SAF RC and RACF return code is presented as RACF RC in the following section. |

SAF RC
Meaning
00 RACROUTE REQUEST=TOKENBLD has completed successfully.

RACF RC
Meaning
08 Indicates REQUEST=TOKENBLD has completed successfully.

Reason Code
Meaning
10 TOKNOUT area specified was larger than expected; on return the token-length field contains the expected length.
14 STOKEN area specified was larger than expected.
20 TOKNIN area specified was larger than expected.
08 The requested function failed.

RACF RC
Meaning
00 An error occurred before the function could initiate.

Reason Code
Meaning
00 A recovery environment could not be established.

Example
The following example shows how a RACROUTE REQUEST=TOKENBLD macro can be specified to replace a security label in an existing token.

```
RACROUTE REQUEST=TOKENBLD,TOKNOUT=TOKOUT,TOKNIN=TOKIN X
SECLBL=SLBL,RELEASE=1.9,WORKA=RACWK
```

```
TOKOUT DS 0CL80
DC XL2'5001' /* FIRST 2 BYTES SPECIFY TOKEN VERSION */
DC XL78'0'
TOKIN DS 0CL80
DC XL2'5001'
DC XL78'0'
SLBL DC CL8'INTERNAL'
RACWK DC CL512
```

Note: Additional keywords required by RACF to complete the request, such as WORKA, are specified on RACROUTE itself.
### RACROUTE REQUEST=TOKENBLD (list form)

The list form of the RACROUTE REQUEST=TOKENBLD macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=TOKENBLD macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>parameter</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Symbol. Begin name in column 1.</td>
</tr>
<tr>
<td>RACROUTE</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
<tr>
<td>REQUEST=TOKENBLD</td>
<td><strong>Default:</strong> RELEASE=1.6</td>
</tr>
<tr>
<td>,EXENODE=execution node addr</td>
<td>execution node addr: A-type address</td>
</tr>
<tr>
<td>,GROUP=group addr</td>
<td>group addr: A-type address</td>
</tr>
<tr>
<td>,POE=port of entry addr</td>
<td>port of entry addr: A-type address</td>
</tr>
<tr>
<td>,POENET=network name addr</td>
<td>network name addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,REMOTE=YES,REMOTE=NO</td>
<td><strong>Default:</strong> REMOTE=NO</td>
</tr>
<tr>
<td>,SECLABL=seclabel addr</td>
<td>seclabel addr: A-type address</td>
</tr>
<tr>
<td>,SESSION=type</td>
<td>type: Any valid session type <strong>Default:</strong> SESSION=TSO</td>
</tr>
<tr>
<td>,SGROUP=submitting group addr</td>
<td>submitting group addr: A-type address</td>
</tr>
<tr>
<td>,SNODE=submitting node addr</td>
<td>submitting node addr: A-type address</td>
</tr>
<tr>
<td>,STOKEN=token addr</td>
<td>token addr: A-type address</td>
</tr>
<tr>
<td>,SUSERID=submitting userid addr</td>
<td>submitting userid addr: A-type address</td>
</tr>
<tr>
<td>,TERMID=terminal addr</td>
<td>terminal addr: A-type address</td>
</tr>
<tr>
<td>,TOKNIN=input token addr</td>
<td>input token addr: A-type address</td>
</tr>
<tr>
<td>,TOKNOUT=output token addr</td>
<td>output token addr: A-type address</td>
</tr>
<tr>
<td>,TRUSTED=YES,TRUSTED=NO</td>
<td><strong>Default:</strong> TRUSTED=NO</td>
</tr>
<tr>
<td>,USERID=userid addr</td>
<td>userid addr: A-type address</td>
</tr>
</tbody>
</table>
REQUEST=TOKENBLD

The parameters are explained under the standard form of the RACROUTE REQUEST=TOKENBLD macro instruction with the following exception:

, MF=L

specifies the list form of the RACROUTE REQUEST=TOKENBLD macro instruction.

RACROUTE REQUEST=TOKENBLD (execute form)

The execute form of the RACROUTE REQUEST=TOKENBLD macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=TOKENBLD macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

```
name
```

name: Symbol. Begin name in column 1.

b

One or more blanks must precede RACROUTE.

RACROUTE

b

One or more blanks must follow RACROUTE.

```
REQUEST=TOKENBLD

,RELEASE=number
,RELEASE=<(CHECK)
,RELEASE=(number, CHECK)

,EXENODE=execution
    node addr

,GROUP=group addr

,POE=port of entry addr
,POENET=network name
    addr

,REMOTE=YES
,REMOTE=NO

,SECLABL=seclabel addr

,SESSION=type

,SGROUP=submitting
    group addr

,SNODE=submitting node
    addr

,STOKEN=token addr
```

number: See standard form

Default: RELEASE=1.6

execution node addr: Rx-type address or register (2) – (12)

group addr: Rx-type address or register (2) – (12)

port of entry addr: Rx-type address or register (2) – (12)

network name addr: A-type address or register (2) – (12)

seclabel addr: Rx-type address or register (2) – (12)

type: Any valid session type

submitting group addr: Rx-type address or register (2) – (12)

submitting node addr: Rx-type address or register (2) – (12)

token addr: Rx-type address or register (2) – (12)
### REQUEST=TOKENBLD

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>,USERID=submitting userid addr</td>
<td>submitting userid addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,TERMID=terminal addr</td>
<td>terminal addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,TOKNIN=input token addr</td>
<td>input token addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,TOKNOUT=output token addr</td>
<td>output token addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,TRUSTED=YES</td>
<td></td>
</tr>
<tr>
<td>,TRUSTED=NO</td>
<td></td>
</tr>
<tr>
<td>,USERID=userid addr</td>
<td>userid addr: Rx-type address or register (2) - (12)</td>
</tr>
<tr>
<td>,MF=(E,ctrl addr)</td>
<td>ctrl addr: Rx-type address or register (1) or (2) – (12)</td>
</tr>
</tbody>
</table>

The parameters are explained under the standard form of the RACROUTE REQUEST=TOKENBLD macro instruction with the following exceptions:

- **,RELEASE=number**
- **,RELEASE=(,CHECK)**
- **,RELEASE=(number,CHECK)**

specifies the RACF release level of the parameter list to be generated by this macro.

To use the parameters associated with a release, you must specify the number of that release or a later release number. If you specify an earlier release level, the parameter is not accepted by macro processing, and an error message is issued at assembly time.

Compatibility between the list and execute forms of the RACROUTE REQUEST=TOKENBLD macro is validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.

The default is RELEASE=1.6.

- **,MF=(E,ctrl addr)**

specifies the execute form of the RACROUTE REQUEST=TOKENBLD macro, using a remote, control-program parameter list.

### RACROUTE REQUEST=TOKENBLD (modify form)

The modify form of the RACROUTE REQUEST=TOKENBLD macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=TOKENBLD macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

- **name**: Symbol. Begin name in column 1.

- **b**: One or more blanks must precede RACROUTE.
REQUEST=TOKENBLD

RACROUTE

b

One or more blanks must follow RACROUTE.

REQUEST=TOKENBLD

,EXENODE=execution node
    node addr: Rx-type address or register (2) – (12)

,GROUP=group addr
    group addr: Rx-type address or register (2) – (12)

,POE=port of entry
    port of entry addr: Rx-type address or register (2) – (12)

,POENET=network name
    network name addr: A-type address or register (2) – (12)

,REMOTE=YES
,REMOTE=NO

,SECLAB=seclabel addr
    seclabel addr: Rx-type address or register (2) – (12)

,SESSION=type
    type: Any valid session type

,SGROUP=submitting group addr
    submitting group addr: Rx-type address or register (2) – (12)

,SNODE=submitting node
    submitting node addr: Rx-type address or register (2) – (12)

,STOKEN=token addr
    token addr: Rx-type address or register (2) – (12)

,SUSERID=submitting userid
    submitting userid addr: Rx-type address or register (2) – (12)

,TERMID=terminal addr
    terminal addr: Rx-type address or register (2) – (12)

,TOKENIN=input token
    input token addr: Rx-type address or register (2) – (12)

,TOKENOUT=output token
    output token addr: Rx-type address or register (2) – (12)

,TRUSTED=YES
,TRUSTED=NO

,USERID=userid
    userid addr: Rx-type address or register (2) – (12)

,MF=(M,ctrl addr)
    ctrl addr: Rx-type address or register (1) or (2) – (12)

The parameters are explained under the standard form of the RACROUTE REQUEST=TOKENBLD macro instruction with the following exceptions:

,,MF=(M,ctrl addr)
    specifies the modify form of the RACROUTE REQUEST=TOKENBLD macro, using a remote, control-program parameter list.
RACROUTE REQUEST=TOKENMAP: Access token fields

The RACROUTE REQUEST=TOKENMAP macro maps a token in either internal or external format. Internal format is the encoded data format returned from a RACROUTE REQUEST=VERIFYX or a RACROUTE REQUEST=TOKENXTR. External format is the user-readable format that is mapped by the ICHRUTKN macro. See “RUTKN: Resource/User Security Token” in z/OS Security Server RACF Data Areas. RACROUTE REQUEST=TOKENMAP is the only interface used to map token data.

The primary purpose of the RACROUTE REQUEST=TOKENMAP function is to allow a caller to access individual fields within the UTOKEN. The caller needs to provide the proper length and version for the corresponding format version of a token.

To use this service, you must specify RELEASE=1.9 or a later release number.

This function is performed by SAF.

The caller cannot hold locks when issuing RACROUTE REQUEST=TOKENMAP.

RACROUTE REQUEST=TOKENMAP (standard form)

The standard form of the RACROUTE REQUEST=TOKENMAP macro is written as follows. For a description of additional keywords that you can code and additional parameters that are required on the RACROUTE request, but that are not specific to this request type, see “RACROUTE (standard form)” on page 13.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Symbol. Begin name in column 1.</td>
</tr>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
<tr>
<td>RACROUTE</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>One or more blanks must follow RACROUTE.</td>
</tr>
<tr>
<td>REQUEST=TOKENMAP</td>
<td></td>
</tr>
<tr>
<td>,TOKNIN=input token addr</td>
<td>input token addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,TOKNOUT=output token addr</td>
<td>output token addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,FORMOUT=INTERNAL,FORMOUT=EXTERNAL</td>
<td>Default: FORMOUT=EXTERNAL</td>
</tr>
<tr>
<td>,XMREQ=YES, ,XMREQ=NO</td>
<td>Default: NO</td>
</tr>
<tr>
<td>,MF=S</td>
<td></td>
</tr>
</tbody>
</table>

The parameters are explained as follows:

,FORMOUT=EXTERNAL
REQUEST=TOKENMAP

,,FORMOUT=INTERNAL
   specifies the format of the output token area.

   INTERNAL
      This is the encoded data format that is returned from a RACROUTE
      REQUEST=VERIFYX, RACROUTE REQUEST=TOKENXTR or a
      RACROUTE REQUEST=TOKENBLD.

   EXTERNAL
      The user-readable format is that which is mapped by the ICHRUTKN
      macro. See “RUTKN: Resource/User Security Token” in z/OS Security
      Server RACF Data Areas.

,,TOKNIN=input token addr
   specifies the address of the UTOKEN or RTOKEN that is to be converted to
   internal or external format.

,,TOKNOUT=output token address
   specifies the address of the caller-provided area for the converted token data.
   The first byte of storage at the address specified must contain the token length.
   The second byte must contain the format version of the token. This provides
   for downward compatibility with all versions of the token map.

,,XMREQ=NO
,,XMREQ=YES
   indicates whether or not the caller is in cross-memory mode.

      NO      Indicates that the caller is not in cross-memory mode. (NO is the
              default.)

      YES     Signifies that the caller can (but need not) be running in cross-memory
              mode.

   The caller must be in primary ASC mode when RACROUTE is issued.

,,MF=S
   specifies the standard form of the RACROUTE REQUEST=TOKENMAP macro
   instruction.

Return codes and reason codes
When you execute the macro, space for the RACF return code and reason code is
reserved in the first two words of the RACROUTE parameter list. You can access
them using the ICHSAFP mapping macro, by loading the ICHSAFP pointer with
the label that you specified on the list form of the macro. When control is returned,
register 15 contains the SAF return code.

<table>
<thead>
<tr>
<th>SAF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACROUTE REQUEST=TOKENMAP has completed successfully.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Reason described by the following hexadecimal reason codes:</td>
</tr>
</tbody>
</table>
REQUEST=TOKENMAP

Reason Code
Meaning
00  The request was successful.
04  TOKEN was not converted; already in requested format.
0C  TOKNOUT area too large; token was successfully extracted.

04  RACROUTE REQUEST=TOKENMAP did not complete successfully.

RACF RC
Meaning
00  Reason described by the following hexadecimal reason codes:

Example
The following is an example of invoking the TOKENMAP function:

RACROUTE REQUEST=TOKENMAP,
TOKIN=TOKIN,TOKNOUT=TOKOUT,
RELEASE=1.9,WORKA=RACWK

TOKIN DS 0CL80
DC XL2'5001' /*FIRST 2 BYTES SPECIFY TOKEN VERSION */
DC XL78'0'

TOKOUT DS 0CL80
DC XL2'5001'
DC XL78'0'

RACWK DS CL'512'

RACROUTE REQUEST=TOKENMAP (list form)
The list form of the RACROUTE REQUEST=TOKENMAP macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=TOKENMAP macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
<tr>
<td>RACROUTE</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>One or more blanks must follow RACROUTE.</td>
</tr>
</tbody>
</table>

REQUEST=TOKENMAP
.FORMOUT=INTERNAL
.FORMOUT=EXTERNAL

.TOKIN=input token addr: A-type address
REQUEST=TOKENMAP

,TOKNOUT=\output{token addr} \output{addr} 
\output{ addr} \output{ output token addr: A-type address}

,XMREQ=YES \Default{NO}
,XMREQ=NO
,\MF=L

The parameters are explained under the standard form of the RACROUTE REQUEST=TOKENMAP macro with the following exception:

,\MF=L
specifies the list form of the RACROUTE REQUEST=TOKENMAP macro instruction.

RACROUTE REQUEST=TOKENMAP (execute form)

The execute form of the RACROUTE REQUEST=TOKENMAP macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=TOKENMAP macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

name

\textit{name}: Symbol. Begin \textit{name} in column 1.

b

One or more blanks must precede RACROUTE.

RACROUTE

b

One or more blanks must follow RACROUTE.

REQUEST=TOKENMAP

,\FORMOUT=INTERNAL
,\FORMOUT=EXTERNAL

,\TOKNIN=\textit{input token addr} \textit{input token addr}: Rx-type address or register (2) – (12)

,\TOKNOUT=\textit{output token addr} \textit{output token addr}: Rx-type address or register (2) – (12)

,XMREQ=YES
,XMREQ=NO

,\MF=(E,\textit{ctrl addr}) \textit{ctrl addr}: Rx-type address or register (1) or (2) – (12)

The parameters are explained under the standard form of the RACROUTE REQUEST=TOKENMAP macro with the following exception:

,\MF=(E,\textit{ctrl addr})
specifies the execute form of the RACROUTE REQUEST=TOKENMAP macro instruction.
RACROUTE REQUEST=TOKENMAP (modify form)

The modify form of the RACROUTE REQUEST=TOKENMAP macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=TOKENMAP macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

```
name name: Symbol. Begin name in column 1.

b One or more blanks must precede RACROUTE.

RACROUTE

b One or more blanks must follow RACROUTE.

REQUEST=TOKENMAP

,FORMOUT=INTERNAL
,FORMOUT=EXTERNAL

,TOKNIN=input token addr
input token addr: Rx-type address or register (2) – (12)

,TOKNOUT=output token addr
output token addr: Rx-type address or register (2) – (12)

,XMREQ=YES.
,XMREQ=NO

,MF=(M,ctrl addr)
ctrl addr: Rx-type address or register (1) or (2) – (12)
```

The parameters are explained under the standard form of the RACROUTE macro with the following exception:

```
,MF=(M,ctrl addr)
```

specifies the modify form of the RACROUTE macro instruction.

RACROUTE REQUEST=TOKENXTR: Extract U TOKENs

The RACROUTE REQUEST=TOKENXTR macro extracts a UTOKEN from the current address space, task, or a caller-specified ACEE. Any information not available from the ACEE is returned as blanks, or is defaulted. The ICHRUTKN macro maps the UTOKEN. See “RUTKN: Resource/User Security Token” in z/OS Security Server RACF Data Areas.

To use this service, you must specify RELEASE=1.9 or a later release number.

This function is performed by SAF.

The caller cannot hold any locks when issuing RACROUTE REQUEST=TOKENXTR.
REQUEST=tokenXTR

RACROUTE REQUEST=tokenXTR (standard form)

The standard form of the RACROUTE REQUEST=tokenXTR macro is written as follows. For a description of additional keywords that you can code and additional parameters that are required on the RACROUTE request, but that are not specific to this request type, see "RACROUTE (standard form)" on page 13.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
</tbody>
</table>

RACROUTE

| b    | One or more blanks must follow RACROUTE. |

REQUEST=tokenXTR

!,tokenOut=return token addr output token addr: A-type address or register (2) – (12)
!,ACEE=acee addr acee addr: A-type address or register (2) - (12)
!,XMREQ=YES Default: XMREQ=NO
!,XMREQ=NO
!,MF=S

The parameters are explained as follows:

!,ACEE=acee addr

specifies the address of the ACEE from which information is to be extracted. If you do not specify the ACEE keyword, TOKENXTR extracts the information it needs from the TCBSENV field of the task control block if it is nonzero; if it is zero, TOKENXTR extracts information it needs from the ASXBSENV field.

**Note:** If no ACEE is specified or found, a token is returned with the following information:

- The user ID is ‘*’.
- A default TOKEN flag is on.
- An undefined user flag is on.
- A flag indicating this token is created for a pre-RACF 1.9 system.

If there is a down-level ACEE, a token is returned with indication of a pre-RACF 1.9 system; certain fields might be blank or zero, such as SECLABEL, submitting user ID, and other information not available for a down-level ACEE.

**Restriction:** For a nested ACEE, token information is extracted only from the primary identity, not from the embedded daemon identity.

!,tokenOut=return token addr

specifies the address where the requester wants TOKENXTR to return the UTOKEN that was extracted from the ACEE. The first byte of storage at the address specified must contain the number of bytes of available storage. The second byte must contain the format version of the token.
REQUEST=TOKENXTR

\^XMREQ=NO
\^XMREQ=YES

indicates whether or not the caller is in cross-memory mode.

NO Indicates that the caller is not in cross-memory mode. (NO is the default.)

YES Signifies that the caller might (but need not) be running in cross-memory mode.

The caller must be in primary ASC mode when RACROUTE is issued.

\^MF=S

specifies the standard form of the RACROUTE REQUEST=TOKENXTR macro instruction.

Return codes and reason codes

When you execute the macro, space for the RACF return code and reason code is reserved in the first two words of the RACROUTE parameter list. You can access them using the ICHSAFP mapping macro by loading the ICHSAFP pointer with the label that you specified on the list form of the macro. When control is returned, register 15 contains the SAF return code.

<table>
<thead>
<tr>
<th>SAF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACROUTE REQUEST=TOKENXTR has completed successfully.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Reason described by the following hex reason codes:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>The request was successful.</td>
</tr>
<tr>
<td>04</td>
<td>ACEE supplied is not valid (down level). Information is defaulted if it could not be extracted.</td>
</tr>
<tr>
<td>08</td>
<td>No ACEE available. Information is defaulted if it could not be extracted.</td>
</tr>
<tr>
<td>0C</td>
<td>TOKNOUT area length was too large.</td>
</tr>
<tr>
<td>04</td>
<td>RACROUTE REQUEST=TOKENXTR did not complete successfully.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Reason described by the following hex reason codes:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>XMREQ=YES was specified in a non-MVS/ESA environment.</td>
</tr>
</tbody>
</table>
REQUEST=TOKENXTR

Example
This example shows how to extract information from the ACEE to determine which type of label to put on printed output data. See the example section for TOKENMAP for an example of converting the token returned by TOKENXTR into readable form.

RACROUTE REQUEST=TOKENXTR,TOKNOUT=TOKOUT,
  WORKA=RACWK,
  RELEASE=1.9

...  

RACWK DS CL512
TOKOUT DS 0CL80
  DC XL2'5001' /*FIRST 2 BYTES SPECIFY THE TOKEN VERSION */
  DC XL78'0'

RACROUTE REQUEST=TOKENXTR (list form)
The list form of the RACROUTE REQUEST=TOKENXTR macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=TOKENXTR macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
<tr>
<td>RACROUTE</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>One or more blanks must follow RACROUTE.</td>
</tr>
</tbody>
</table>

REQUEST=TOKENXTR
  ,ACEE=acee addr  acee addr: A-type address
  ,TOKNOUT=output token addr  output token addr: A-type address
  ,XMREQ=YES Default: XMREQ=NO
  ,XMREQ=NO
  ,MF=L

The parameters are explained under the standard form of the RACROUTE REQUEST=TOKENXTR macro with the following exception:

,MF=L
  specifies the list form of the RACROUTE REQUEST=TOKENXTR macro instruction.

RACROUTE REQUEST=TOKENXTR (execute form)
The execute form of the RACROUTE REQUEST=TOKENXTR macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=TOKENXTR macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.
REQUEST=TOKENXTR

\[\text{name: Symbol. Begin name in column 1.}\]

\[\text{b: One or more blanks must precede RACROUTE.}\]

RACROUTE

\[\text{b: One or more blanks must follow RACROUTE.}\]

\[
\begin{align*}
\text{REQUEST=TOKENXTR} \\
\text{,ACEE=acee addr} & \quad \text{acee addr: Rx-type address or register (2) - (12)} \\
\text{,TOKNOUT=output token addr} & \quad \text{output token addr: Rx-type address or register (2) – (12)} \\
\text{,XMREQ=NO} \\
\text{,XMREQ=YES} \\
\text{,MF=(E, ctrl addr)} & \quad \text{ctrl addr: Rx-type address or register (1) - (12)}
\end{align*}
\]

The parameters are explained under the standard form of the RACROUTE REQUEST=TOKENXTR macro with the following exception:

\[\text{,MF=(E, ctrl addr)}\]

specifies the execute form of the RACROUTE REQUEST=TOKENXTR macro instruction.

**RACROUTE REQUEST=TOKENXTR (modify form)**

The modify form of the RACROUTE REQUEST=TOKENXTR macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=TOKENXTR macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

\[\text{name: Symbol. Begin name in column 1.}\]

\[\text{b: One or more blanks must precede RACROUTE.}\]

RACROUTE

\[\text{b: One or more blanks must follow RACROUTE.}\]

\[
\begin{align*}
\text{REQUEST=TOKENXTR} \\
\text{,ACEE=acee addr} & \quad \text{acee addr: Rx-type address or register (2) - (12)} \\
\text{,TOKNOUT=output token addr} & \quad \text{output token addr: Rx-type address or register (2) – (12)} \\
\text{,XMREQ=YES} \\
\text{,XMREQ=NO} \\
\text{,MF=M}
\end{align*}
\]
REQUEST=TOKENXTR

The parameters are explained under the standard form of the RACROUTE REQUEST=TOKENXTR macro with the following exception:

```
,FM=M
```
specifies the modify form of the RACROUTE REQUEST=TOKENXTR macro instruction.

**RACROUTE REQUEST=VERIFY: Identify and verify a RACF-defined user**

The RACROUTE REQUEST=VERIFY macro provides RACF user identification and verification. The macro instruction identifies a user and verifies that the user is defined to RACF and has done either of the following tasks:

- He has supplied a valid password phrase, or
- He has supplied a valid password or operator-identification card (OIDCARD parameter) or both.

You can protect applications by using profiles in the APPL class along with this macro to control the users able to use applications. For more information about protecting applications, see [z/OS Security Server RACF Security Administrator's Guide](#).

A subsystem can use REQUEST=VERIFY and the new keywords to create an ACEE. (For information on VLF considerations for ACEEs, see [z/OS Security Server RACF System Programmer's Guide](#).) If RACF is not active or not installed, SAF, using the new keywords, builds a default ACEE to satisfy the request. The purpose of this use of REQUEST=VERIFY is for job submission and user verification checking.

The following order of priority exists for replacing the fields in the existing TOKEN:

- Keywords specified on the request take precedence over corresponding fields in the TOKNIN and STOKEN parameters.
- All fields within the token specified by the TOKNIN keyword take precedence over those specified by STOKEN.
- The fields for the submitter's ID, submitter's group, submit node, execution node, session, port of entry and its class, as obtained from the token specified by the STOKEN keyword are last.

If you do not want certain fields overridden, do not specify keywords for those fields.

To issue the RACROUTE REQUEST=VERIFY macro, the calling module must be authorized (APF-authorized, in system key 0–7, or in supervisor state) or the NEWPASS, PHRASE, NEWPHRASE, ICRX, ICTX, and IDID keywords must be omitted and the calling module must be in the RACF-authorized caller table and fetched from an authorized library and reentrant.

However, the caller does not need to be authorized or in the RACF-authorized caller table when specifying ENVIR=VERIFY on the RACROUTE REQUEST=VERIFY macro.
Guideline: If you run programs that issue the RACROUTE REQUEST=VERIFY macro, run those programs APF-authorized. See z/OS Security Server RACF System Programmer’s Guide for information on the authorized caller table.

The caller cannot hold any locks when issuing RACROUTE REQUEST=VERIFY.

Automatic direction of application updates does not propagate RACROUTE REQUEST=VERIFY requests that update the RACF database; however, the following ICHEINTY requests issued by RACROUTE REQUEST=VERIFY are propagated:

- The ICHEINTY setting the revoke flag in the user profile when a user is being revoked due to inactivity or incorrect password attempts.
- The ICHEINTY increasing the revoke count when a user enters an incorrect password.
- The ICHEINTY that resets the revoke count to 0 when a user enters a valid password.

The ICHEINTY issued by RACROUTE REQUEST=VERIFY to change the password in the user’s profile is not eligible for propagation by automatic direction of application updates because it is already eligible for propagation by automatic password direction.

Note:

1. Unless the caller specifies the ACEE= parameter on a RACROUTE REQUEST=VERIFY,ENVIR=CREATE macro instruction, the ACEE is always placed below 16MB.
2. If the caller specifies the ACEE= parameter, is executing in 31-bit addressing mode, and does not specify LOC=BELOW on the RACROUTE macro instruction, the ACEE is placed, if possible, above 16MB.
3. Some functions (such as EXTRACT) require that an ACEE exist. RACF supports use of the *BYPASS* user ID to create an ACEE to satisfy these cases when no other ACEE is available.

To create this environment, specify the user ID as *BYPASS*, specify PASSCHK=NO, and do not specify a password field at all. A successful use is not audited but a failure may be audited if requested.

If an ACEE with a user ID of *BYPASS* is used with REQUEST=AUTH, a return code of 4 is returned. Depending upon the application, this may allow access to a resource.

---

**RACROUTE REQUEST=VERIFY (standard form)**

The standard form of the RACROUTE REQUEST=VERIFY macro is written as follows. For a description of additional keywords that you can code and additional parameters that are required on the RACROUTE request, but that are not specific to this request type, see “RACROUTE (standard form)” on page 13.

Note:

1. Application programs must be structured so that a task requesting RACF services does not do so while other I/O initiated by the task is outstanding. If such I/O is required, the task should either wait for the other I/O to complete before requesting RACF services, or the other I/O should be initiated under a separate task. This is necessary to assure proper processing in recovery situations.
2. The most common way to create an ACEE is to issue a RACROUTE REQUEST=VERIFY, specifying ENVIR=CREATE. After all RACROUTE invocations are complete, the invoker should issue RACROUTE REQUEST=VERIFY with ENVIR=DELETE specified to delete the ACEE previously created.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
</tbody>
</table>

RACROUTE

| b | One or more blanks must follow RACROUTE. |

REQUEST=VERIFY

| ,ACEE=address of fullword | address of fullword: A-type address or register (2) – (12) |
| ,ACTINFO=account addr | account addr: A-type address or register (2) – (12) |
| ,APPL='applname' | applname: 1–8 character name |
| ,APPL=applname addr | applname addr: A-type address or register (2) – (12) |
| ,ENCRYPT=YES | Default: ENCRYPT=YES |
| ,ENCRYPT=NO | |
| ,ENVIR=CREATE | Default: ENVIR=CREATE |
| ,ENVIR=VERIFY | |
| ,ENVIR=CHANGE | |
| ,ENVIR=DELETE | |
| ,ENVRIN=envr data addr | envr data addr: A-type address or register (2) – (12) |
| ,ENVROUT=envr data addr | envr data addr: A-type address or register (2) – (12) |
| ,ERROROPT=ABEND | Default: ERROROPT=ABEND |
| ,ERROROPT=NOABEND | |
| ,EXENODE=execution node addr | execution node addr: A-type address or register (2) – (12) |
| ,GROUP=group addr | group addr: A-type address or register (2) – (12) |
| ,ICRX=icrx addr | icrx addr: A-type address or register (2) – (12) |
| ,ICTX=ictx addr | ictx addr: A-type address or register (2) – (12) |
| ,IDID=idid addr | idid addr: A-type address or register (2) – (12) |
| ,INSTLN=parm list addr | parm list addr: A-type address or register (2) – (12) |
| ,JOBNAME=jobname addr | jobname addr: A-type address or register (2) – (12) |
| ,LOC=BELOW | Default: See parameter description. |
| ,LOC=ANY | |
REQUEST=VERIFY

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOG=ASIS</td>
<td>Default: LOG=ASIS</td>
</tr>
<tr>
<td>LOG=ALL</td>
<td>LOG=NONE</td>
</tr>
<tr>
<td>LOGSTR=logstr addr</td>
<td>logstr addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>NESTED=YES</td>
<td>Default: NESTED=NO</td>
</tr>
<tr>
<td>NESTED=NO</td>
<td>NESTED=COPY</td>
</tr>
<tr>
<td>NEWPASS=new password addr</td>
<td>new password addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>NEWPHRASE=new password phrase addr</td>
<td>new password phrase addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>OIDCARD=oid addr</td>
<td>oid addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>PASSCHK=YES</td>
<td>Default: PASSCHK=YES</td>
</tr>
<tr>
<td>PASSCHK=NO</td>
<td>PASSWRD=password addr</td>
</tr>
<tr>
<td>PASSWRD=password addr</td>
<td>password addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>PGMNAME=programmer name addr</td>
<td>programmer name addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>PHRASE=password phrase addr</td>
<td>password phrase addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>POE=port of entry addr</td>
<td>port of entry addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>POENET=network name addr</td>
<td>network name addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>REMOTE=YES</td>
<td>Default: REMOTE=NO</td>
</tr>
<tr>
<td>REMOTE=NO</td>
<td>REMOTE</td>
</tr>
<tr>
<td>SECLABL=seclabel addr</td>
<td>seclabel addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>SERVAUTH=sercauth addr</td>
<td>sercauth addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>SESSION=type</td>
<td>type: Any valid session type</td>
</tr>
<tr>
<td></td>
<td>Default: SESSION=TSO</td>
</tr>
<tr>
<td>SGROUP=submitting group addr</td>
<td>submitting group addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>SMC=YES</td>
<td>Default: SMC=YES</td>
</tr>
<tr>
<td>SMC=NO</td>
<td>SMC=NO</td>
</tr>
<tr>
<td>SNODE=submitting node addr</td>
<td>submitting node addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>START=procname addr</td>
<td>procname addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>STAT=ASIS</td>
<td>Default: STAT=ASIS</td>
</tr>
<tr>
<td>STAT=NO</td>
<td>STAT=NO</td>
</tr>
</tbody>
</table>
REQUEST=VERIFY

- \_STOKEN= stoken addr: A-type address or register (2) – (12)
- \_SUBPOOL= subpool number: Decimal digit 0–255
- \_USERID= submitting userid addr: A-type address or register (2) – (12)
- \_SYSTEM= NO or YES
- \_USERID= userid addr
- \_X500NAME= X500 name pair addr
- \_TRUSTED= YES or NO
- \_TERMID= terminal addr
- \_TOKNIN= utoken addr
- \_TOKNOUT= utoken addr
- \_MF= S

The parameters are explained as follows:

- **\_ACEE= address of fullword**
  - Specifies the address of a fullword to be used as described below.

  **ENVIR=DELETE** specifies the address of a fullword that contains the address of the ACEE to be deleted. If ACEE= is not specified or ACEE=0 is specified, and the TCBSENV field for the task using the RACROUTE REQUEST=VERIFY is nonzero, the ACEE pointed to by the TCBSENV is deleted, and TCBSENV is set to zero. If the TCBSENV and ASXBSENV fields both point to the same ACEE, ASXBSENV is also set to zero. If no ACEE address is passed, and TCBSENV is zero, the ACEE pointed to by ASXBSENV is deleted, and ASXBSENV is set to zero.

  **ENVIR=CHANGE** specifies the address of a fullword that contains the address of the ACEE to be changed. If ACEE= is not specified, and the TCBSENV field for the task using the RACROUTE REQUEST=VERIFY is nonzero, the ACEE pointed to by the TCBSENV is changed. If TCBSENV is 0, the ACEE pointed to by ASXBSENV is changed.

  **ENVIR=CREATE** specifies the address of a fullword in which the RACROUTE REQUEST=VERIFY function places the address of the ACEE created.

The user should not create an ACEE above 16MB and place the address in ASXBSENV or TCBSENV. If an ACEE is not specified, the address of the newly created ACEE is stored in the TCBSENV field of the task control block. If the ASXBSENV field contains binary zeros, the new ACEE address is also stored in the ASXBSENV field of the ASXB. If the ASXBSENV field is nonzero, it is not modified. The TCBSENV field is set unconditionally if ACEE= is not specified.
REQUEST=VERIFY

Note:

1. If you omit USERID, GROUP, and PASSWRD and if you code
   ENVIR=CREATE or if ENVIR=CREATE is used as the default, you receive
   a return code of X’00’ and obtain an ACEE that contains an asterisk (*)
   (X’5C’) in place of the USERID and group name.
2. Specifying the ACEE= keyword prevents messages ICH70001I and
   ICH70002I from being issued.
3. When the ASXBSENV field is non-zero and only the new ACEE is being
   anchored in TCBSENV, if the MLACTIVE option is on, the security label
   associated with the address space must be equivalent to the security label
   of the new ACEE.

,,ACTINFO=account addr
specifies the address of a field containing accounting information. This
144-byte area is passed to the RACINIT installation exit routine; it is not used
by the RACROUTE REQUEST=VERIFY routine. The accounting field, if
supplied, should have the following format:
  • The first byte of the field contains the number (in binary) of accounting
    fields.
  • The following bytes contain accounting fields, where each entry for an
    accounting field contains a 1-byte length field, followed by the data.

,,APPL=’applname’
,,APPL=applname addr
specifies the name of the application issuing the RACROUTE
REQUEST=VERIFY to verify the user’s authority to access the application. This
saves the application from having to do a separate RACROUTE
REQUEST=AUTH.

If an address is specified, the address must point to an 8-byte application
name, left-justified, and padded with blanks if necessary.

,,ENCRYPT=YES
,,ENCRYPT=NO
specifies whether RACROUTE REQUEST=VERIFY encodes the old password,
the new password, and the OIDCARD data passed to it.

The default is YES.

YES  Signifies that the data specified by the PASSWRD, NEWPASS, and
      OIDCARD keywords are not pre-encoded. RACROUTE
      REQUEST=VERIFY encodes the data before storing it in the user
      profile or using it to compare against stored data.

NO   Signifies that the data specified by the PASSWRD, NEWPASS, and
      OIDCARD keywords are already encoded. RACROUTE
      REQUEST=VERIFY bypasses the encoding of this data before storing it
      in or comparing it against the user profile.

Note: If the password was shipped from another system, the
encryption method must be the same on all systems using the
password. For example, the RACF password authentication exit,
ICHDEX01, must be identical on all systems.

Note: ENCRYPT=NO does not apply to PHRASE and NEWPHRASE and is
ignored if specified.

,,ENVIR=CREATE
,,ENVIR=VERIFY
REQUEST=VERIFY

,ENVIR=CHANGE,ENVIR=DELETE
specifies the action to be performed by the user initialization component regarding the ACEE.

The default is CREATE.

CREATE
The user should be verified and an ACEE created.

VERIFY
This call is not processed by RACF. However, it can be processed by the SAF installation exit (ICHRTX00) if wanted. If the installation does not satisfy this request in the exit, the RACROUTE caller receives a return code of 4, with RACF return and reason codes of zero. Other security products can choose to process this call. Refer to your security product’s documentation for details.

Note: Because a RACROUTE REQUEST=VERIFY,ENVIR=VERIFY can be issued from a non-authorized state, the SAF installation exit, in this case, runs in a non-authorized state. If the exit invokes a service that requires the issuer to be in an authorized state, it fails.

CHANGE
The ACEE should be modified according to other parameters specified on RACROUTE REQUEST=VERIFY. You can change only the connect group with this option.

DELETE
The ACEE should be deleted. This parameter should be used only if a previous RACROUTE REQUEST=VERIFY has completed successfully.

Guideline: Issue a RACROUTE REQUEST=VERIFY,ENVIR=DELETE to delete only an ACEE that you created. See “Guidelines for changing or deleting an ACEE” on page 232 for other options.

Restriction: ENVIR=CHANGE and ENVIR=DELETE cannot be specified with the parameters as identified in the following table.

<table>
<thead>
<tr>
<th>Restricted parameters</th>
<th>ENVIR=CHANGE</th>
<th>ENVIR=DELETE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTINFO=</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>APPL=</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENVRIN=</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENVROUT=</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ERROROPT=</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>EXENODE=</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>GROUP=</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>ICRX=</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ICTX=</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IDID=</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>JOBNAME=</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NESTED=</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEWPASS=</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NEWPHRASE=</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
\texttt{ENVRIN=} \textit{envr data addr} specifies the data structure that contains the information necessary to re-create a security environment.

The address points to a data structure defined in Table 11 on page 215. The data structure describes the storage location for the ENVR object. While the format of the data structure pointed to by ENVRIN is known to the RACROUTE invokers, the content of the object itself is known only to the external security product.

\textbf{Restrictions:} This keyword is recognized only when SYSTEM=YES and ENVR=CREATE are also specified. In addition, when ENVRIN is specified, only the following keywords are recognized:

- ACEE
- SUBPOOL
- LOC
- TOKNOUT
- TERMID
- POE

Most RACROUTE REQUEST=VERIFY parameters are not recognized in ENVRIN processing because they do not affect the security environment created. The security environment is created based upon the information contained in the ENVR object. No security environment verification (such as GROUP checking or APPL checking) is performed. The ACEE, SUBPOOL, LOC, and TOKNOUT keywords are recognized as output parameters, and are used to specify the location of the security environment and to return...
REQUEST=VERIFY

information about the security environment. The ACEE that results from expanding the ENVR object includes the X500 name, USP, IDID, or ICTX if they exist.

The TERMID and POE keywords are used to set/reset the terminal ID address field within the ACEE, ACEETRMP. If the original request used to create the ENVR specified the TERMID keyword, or the POE keyword with a session type of TSO, the ACEETRMP field set is based upon the keyword information. Since the ENVR can be used across address spaces or jobsteps, the issuer of the request must determine whether the data pointed to by the ACEETRMP field when the ENVR is processed is still valid.

The requester can use the TERMID or POE keyword (or both) to modify the ACEETRMP field to ensure its validity. If the TERMID keyword is specified on an ENVRIN request, its address is placed in the ACEETRMP field. If the POE keyword is specified on the ENVRIN request and the port of entry class associated with the ENVR is the TERMINAL class, then its address is placed in the ACEETRMP field (overriding the TERMID address if TERMID was also specified). No class validation (such as checking the TERMID value against the TERMINAL class) is performed against these keyword values during ENVRIN processing.

ENVROUT=envr data addr

Specifies the data structure that is to hold the information that is used to describe the security environment that was just created.

This information can be used with the ENVRIN keyword to later re-create the security environment without causing I/O to the RACF database. The address points to a data structure defined in Table 11 on page 215. The data structure describes the storage location for the ENVR object. The key of the ENVR data structure is a single-byte value that represents the associated ENVR object storage area. The low-order nibble of this value is the storage key. A key value of X'07' returns an ENVR object in key 07 storage.

While the format of the data structure pointed to by ENVROUT is known to the RACROUTE invokers, the content of the object itself is determined by the external security product.

Restriction: This keyword is only recognized when SYSTEM=YES and ENVIR=CREATE are also specified.

Figure 5 on page 215 and Table 11 on page 215 represent the ENVR data structure for use with the ENVRIN and ENVROUT keywords. The data structure must start on a fullword boundary.
REQUEST=VERIFY

<table>
<thead>
<tr>
<th>Description</th>
<th>Length (bytes)</th>
<th>ENVROUT usage</th>
<th>ENVRIN usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENVR object length</td>
<td>4</td>
<td>Output</td>
<td>Input</td>
</tr>
<tr>
<td>ENVR object storage area length</td>
<td>4</td>
<td>Input/output</td>
<td>Input</td>
</tr>
<tr>
<td>ENVR object storage area address</td>
<td>4</td>
<td>Input/output</td>
<td>Input</td>
</tr>
<tr>
<td>ENVR object storage area subpool</td>
<td>1</td>
<td>Input</td>
<td>N/A</td>
</tr>
<tr>
<td>ENVR object storage area key</td>
<td>1</td>
<td>Input</td>
<td>N/A</td>
</tr>
</tbody>
</table>

The ENVR object storage area can be supplied by the caller or obtained by RACF. If supplied by the caller, it must be on a doubleword boundary and be associated with the job step task. If RACF obtains the storage area, it is on a doubleword boundary and is associated with the jobstep task. The storage is allocated based on the mode of the caller (LOC=ANY for 31-bit callers and LOC=BELLOW for 24-bit callers). The following table shows how the field values affect ENVROUT processing.

<table>
<thead>
<tr>
<th>ENVR object storage area length</th>
<th>ENVR object storage area address</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero</td>
<td>Any value</td>
<td>RACF obtains minimum storage size needed to contain the ENVR object. Storage size is returned in the ENVR object storage area length. Storage address is returned in the ENVR object storage area address.</td>
</tr>
<tr>
<td>Nonzero</td>
<td>Zero</td>
<td>RACF obtains storage size specified or the minimum storage size needed to contain the ENVR object. Storage size is returned in the ENVR object storage area length. Storage address is returned in the ENVR object storage area address.</td>
</tr>
</tbody>
</table>

Figure 5. ENVR data structure
REQUEST=VERIFY

Table 12. ENVROUT storage area processing (continued)

<table>
<thead>
<tr>
<th>ENVR object storage area length</th>
<th>ENVR object storage area address</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nonzero</td>
<td>Nonzero</td>
<td>RACF attempts to use the storage area provided. If the area is too small to contain the ENVR object, RACF frees the storage area provided and obtains the minimum storage size needed to contain the ENVR object. Storage size is returned in the ENVR object storage area length. Storage address is returned in the ENVR object storage area address.</td>
</tr>
</tbody>
</table>

Storage is obtained and freed in the subpool and key specified in the ENVR data structure.

Since the ENVR object length is returned to the caller, the ENVR object can be moved from one storage area to another. This value is supplied as an output to the caller. RACF does not attempt to use this value in either ENVROUT or ENVRIN processing.

The caller is responsible for freeing the ENVR object storage area when it is no longer needed. The length, address, subpool, and key to be used when doing the FREEMAIN are contained in the ENVR data structure.

The ENVR object returned by ENVROUT= is a relocatable object that can be copied from one storage location to another. The returned ENVR object, or a copy of the returned ENVR object, can be specified as input to the RACROUTE interface using the ENVRIN keyword, or to the initACEE callable service using the ENVR_in parameter.

The ENVR object can be passed to other systems, but this should be done with great care. The ENVR object should not be saved for a long time before being used as ENVRIN, and it should not be passed to systems that have different security information. The other systems should share the RACF database and have compatible RACF installation exits and class descriptor tables.

,ERROROPT=ABEND
,ERROROPT=NOABEND

specifies whether RACROUTE REQUEST=VERIFY will abend or not when an error occurs while it is accessing the RACF database.

The default is ABEND.

ABEND

When RACROUTE REQUEST=VERIFY encounters an error accessing the RACF database, issue a 483 abend.

NOABEND

When RACROUTE REQUEST=VERIFY encounters an error accessing the RACF database, 483 abends are suppressed. Instead, the request receives a SAF RC 8, RACF RC X'5C' and a RACF reason code of X'0483yyyy' where 'yyyy' is the RACF manager return code associated with the abend that would have been issued. If you are specifying the ERROROPT keyword with a specific release value, RELEASE=value, Table 13 on page 217 shows how RELEASE= values affect the operation of the ERROROPT keyword.
REQUEST=VERIFY

Table 13. Relationship between the ERROROPT keyword and RELEASE= values

<table>
<thead>
<tr>
<th>Release</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>All earlier releases</td>
<td>ERROROPT keyword is flagged as an unknown keyword.</td>
</tr>
<tr>
<td>7703 and 7705</td>
<td>ERROROPT keyword is syntax checked only and an informational MNOTE indicating that the ERROROPT keyword is being ignored is returned. No abend suppression is performed. However, the SAF parameter list is built with the ERROROPT bit set. This allows programs which are assembled with RELEASE=7703 and RELEASE=7705 to take advantage of ERROROPT=NOABEND once the applications are executed in a z/OS Version 1 Release 3 (HBB7706) or later environment.</td>
</tr>
<tr>
<td>7706 and later</td>
<td>483 abends are replaced with a SAF return code of 8, a RACF return code of X'5C', and a RACF reason code of X'0483yyyy'. &quot;yyyy&quot; is the RACF manager return code associated with the abend that would have been issued.</td>
</tr>
</tbody>
</table>

,EXENODE=execution node addr
specifies the address of an area that contains a 1-byte length field followed by the name of the node on which the unit of work is to be executed. The node name cannot exceed eight bytes.

,GROUP=group addr
specifies the group specified by the user who has entered the system. The address points to a 1-byte length field, followed by the group name, which can be up to eight characters. When the GROUP= keyword is omitted, if a user ID is specified, it defaults to the user's default group; if the user ID is allowed to default to "*", then the group will also default to "*". Applications should fold the group name to uppercase.

,ICRX=icrx addr
specifies an address that points to a caller-provided ICRX area. See the IRRPICRX mapping in "z/OS Security Server RACF Data Areas".

The ICRX might contain an identity context reference (ICR). When an ICRX with an ICR is specified on a RACROUTE REQUEST=VERIFY, ENVIR=CREATE request, VERIFY uses it to determine the appropriate RACF user ID. In this case, all keywords except the following are ignored:
- ACEE
- SUBPOOL
- LOC

RACF attempts to resolve the ICR from the local identity context cache using the R_cacheserv callable service. RACF continues according to one of the following cases:
- If the ICRX contains an ICR and the ICR is resolved, VERIFY retrieves an ENVR object for the user. The ENVR object is used to create an ACEE for the caller in a way that is similar to how the ENVRIN parameter is specified on RACROUTE REQUEST=VERIFY. In this case, the IDID within the ICRX is ignored and reverification of information is not performed.

Note: When creating an ACEE using an ENVR object, the ENVR object might already contain an IDID.
REQUEST=VERIFY

- If the ICRX contains an ICR but the ICR cannot be resolved and if section 2 of the IDID, which is reserved for exclusive use by the External Security Manager, specify a specific user ID, then VERIFY processing continues with this user ID and other security relevant information within section 2 of the IDID. If section 2 of the IDID does not exist or does not specify a RACF user ID, RACROUTE REQUEST=VERIFY fails the request and returns "user not defined".

**Note:** R_cacheserv attempts to resolve the ICR using the local identity context cache and also other relevant identity context caches that it can reach through RACF sysplex communication. Only ICRs that are created by an R_cacheserv store function are supported. See [z/OS Security Server RACF System Programmer’s Guide](https://www.ibm.com/support/knowledgecenter/en/SR1GQ_7.3.0/racsec/racsec_racmap_uses.html) for more information.

If the ICRX does not contain an ICR, and the IDIDMAP class is active, and the RACLIST macro has been issued, RACROUTE REQUEST=VERIFY processing attempts to map to a RACF user ID using the distributed identity information in the IDID and mapping filters previously defined using the RACMAP command.

During the mapping process the following operations are performed on a copy of the data (the original data are not modified):

- All leading and trailing blanks (x'20'), nulls (x'00'), or combination of blanks and null characters are removed from the distributed identity information strings in the IDID, and the lengths are appropriately adjusted.
- If the distributed-identity-user-name (user name) is in X.500 format the name is normalized before it is used to find the matching RACF user ID that is associated with the distributed identity filter.

The normalization rules are described in detail under RACMAP MAP.

If the information in the IDID does not map to a RACF user ID, the RACROUTE REQUEST=VERIFY fails and returns "user not defined".

If a RACF user ID is determined, RACROUTE REQUEST=VERIFY processing continues with this user ID, and PASSCHK=NO is assumed. All other supplied parameters are used. If an ACEE is successfully created, the ACEE points to a copy of the IDID information from the ICRX, and it is used in auditing.

Unless the ICRX was previously obtained from R_cacheserv, the caller must set the ICRX ID, version, subpool, length, and all applicable field length and offset values. RACF validates the id and checks that the specified length values in the RACF sections do not exceed allowed maximum values.

**Note:** The caller must free the ICRX structure.

The ICRX parameter is valid only for the ENVIR=CREATE function of a REQUEST=VERIFY request. The ENVIR=CREATE function ignores the ICRX parameter if both the ENVRIN parameter and SYSTEM=YES parameter are specified.

The ICRX parameter is not valid when specified with any of the following parameters:

- IDID
- ICTX
- NESTED=NO
- NESTED=COPY

When specifying ICRX=, you must specify RELEASE=7760 or later.
REQUEST=VERIFY

\ICTX=ictx addr

specifies an address that points to a caller-provided ICTX area. See the IRRPICTX mapping in [z/OS Security Server RACF Data Areas](#).

The caller must obtain storage for the ICTX above the 16 megabyte line in the ACEE subpool specified by the issuer of RACROUTE REQUEST=VERIFY, or in subpool 255 if an ACEE subpool is not specified. The ICTX area must be on a doubleword boundary and must be associated with the job step task.

The identity context information in the ICTX area is used by RACF when it writes SMF audit records for this user.

The caller is responsible for setting the ICTX id, version, subpool, length, and all applicable field length and offset values. RACF checks the id and version; verifies that the ICTX subpool is the same as the ACEE subpool; validates that the specified length values do not exceed allowed maximum values; then sets ACEEICTX to the address of the caller-provided area. On an unsuccessful return code, the caller is responsible for freeing the input ICTX block. On a successful return code, if the block is used, it is anchored in the ACEE and RACF frees it when the ACEE is deleted. If ICTX= is specified but not used (that is, an ICTX was resolved from an identity context reference), RACF frees the input ICTX block.

The ICTX parameter applies to any SESSION type, but only for ENVIR=CREATE. If it is specified along with an identity context reference (USERID and PASSWRD parameters) that is successfully resolved, RACF builds the ICTX and the caller-provided ICTX area will not be used. When specifying ICTX=, you must also specify RELEASE=7730 or later.

\IDID=idid addr

specifies an address that points to a caller-provided IDID area. See the IRRPIDID mapping in [z/OS Security Server RACF Data Areas](#).

The caller must obtain storage for the IDID above the 16 megabyte line in the ACEE subpool specified by the issuer of RACROUTE REQUEST=VERIFY, or in subpool 255 if an ACEE subpool is not specified.

The IDID area must be on a doubleword boundary and must be associated with the job step task.

The distributed identity information in the IDID area is used by RACF when it writes SMF audit records for this user.

The caller must define the IDID identifier, version, subpool, length, and all applicable field length and offset values. RACF verifies that the IDID subpool is the same as the ACEE subpool, validates the ID, and checks that the specified length values in the RACF sections do not exceed allowed maximum values. RACF then sets ACEEIDID to be the address of the caller-provided area. If RACROUTE REQUEST=VERIFY is not successful, the caller must free the input IDID block. If RACROUTE REQUEST=VERIFY is successful, the caller’s IDID is anchored in the ACEE, and RACF frees it when the ACEE is deleted.

The IDID parameter is valid only for the ENVIR=CREATE function of a REQUEST=VERIFY request. The ENVIR=CREATE function ignores the parameter in the following circumstances:

- If both the ENVRIN parameter and SYSTEM=YES parameter are specified.

**Note:** When creating an ACEE using an ENVR object, the ENVR object might already contain an IDID.
REQUEST=VERIFY

- If a RACROUTE REQUEST=VERIFY, ENVIR=CREATE creates an ACEE for an undefined user, the IDID parameter is ignored.

The IDID parameter is not valid when specified with any of the following parameters:
- ICRX
- ICTX
- NESTED=YES
- NESTED=COPY

When specifying IDID=, you must specify RELEASE=7760 or later.

\INSTLN=parm list addr

specifies the address of an area containing parameter information meaningful to the RACINIT installation exit routine. This area is passed to the installation exit when the exit routine is given control from the RACROUTE REQUEST=VERIFY routine.

The INSTLN parameter can be used by an installation having a user-verification or job-initiation application, and wanting to pass information from one installation module to the RACINIT installation exit routine.

\JOBNAME=jobname addr

specifies the address of the job name of a background job. The address points to an 8-byte area containing the job name (left-justified and padded with blanks if necessary). The JOBNAME parameter is used during authorization checking to verify the user's authority to submit the job. It is passed to the installation exit routine. Also, if \JOBNAME= is specified with the START= parameter, and the STARTED class is active, RACF uses the jobname during its processing to help determine the user ID and group name that are assigned for the started task.

\LOC=BELOW
\LOC=ANY

specifies whether the ACEE and related data areas are to be allocated storage below 16MB (LOC=BELOW), or anywhere (LOC=ANY).

If the ACEE= parameter is not coded, or if the caller is executing in 24-bit mode, LOC=BELOW is the default and LOC=ANY is ignored if specified. In all other cases, the default is LOC=ANY.

\LOG=ASIS
\LOG=ALL
\LOG=NONE

specifies when log records are to be generated.

The default is ASIS.

ASIS Only those requests to create an ACEE that fail generate RACF log records.

ALL A request to create an ACEE, regardless of whether it succeeds or fails, generates a RACF log record. This applies when either ENVIR=CREATE or ENVIR=DELETE is specified.

NONE A request to create an ACEE, regardless of whether it succeeds or fails, does not generate a RACF log record. This applies when either ENVIR=CREATE or ENVIR=DELETE is specified.
REQUEST=VERIFY

LOG=None suppresses both messages and SMF records regardless of MSGSUPP=NO.

Note: SMF records are written for password changes when SETROPTS AUDIT(USER) is in effect regardless of the LOG setting specified.

,LOGSTR=logstr addr specifies the address of a 1-byte length field followed by character data to be written to the system-management-facilities (SMF) data set together with any RACF audit information, if logged.

,NESTED=YES,NESTED=NO,NESTED=COPY specifies whether the created ACEE should contain an ENVR object for the current ACEE of the address space, a copy of the nested ENV object, or neither. The nested security environment can be used in a subsequent RACROUTE REQUEST=FASTAUTH to grant access to a delegated resource (a resource defined with the ‘RACF-DELEGATED’ string in the APPLDATA field) when the client is not specifically permitted to it, but the daemon is authorized. The default is NESTED=NO.

Rule: When specifying NESTED=, you must also specify RELEASE=7720 or later.

YES Specifies that the created ACEE should contain an ENVR object for the ACEE for the current address space, if one exists. The ACEE is created as specified by the other keywords in the request but the ACEE information for the current address space is nested, or encapsulated, within the created ACEE to preserve the current security environment. Preserving the current security environment for a server or daemon address space is useful when a security environment is created for a new client.

NO Specifies that the created ACEE should not contain a nested ENVR object for the ACEE for the current address space.

COPY Specifies that the created ACEE should contain a copy of the ENVR object nested within the ACEE of the current address space, if a nested ACEE exists for the address space. That is, the same security environment that is nested for the address space should also be nested within the newly created ACEE. Server and daemon applications that allow the client to switch identities can use this option to preserve the identity of the server or daemon while building a security environment for a new client identity.

,NEWPASS=new password addr specifies the password that is to replace the user's currently defined password. The address points to a 1-byte length field, followed by the password, which can be up to eight characters.

The NEWPASS= keyword has no effect unless PASSCHK=YES is either defaulted to or explicitly specified and PASSWRD= is also specified. If the NEWPASS= keyword is specified with PASSCHK=NO, no error message is issued but the password is not changed. A new password phrase cannot be set using a password for authentication, nor can a new password be set using a password phrase for authentication.

,NEWPHRASE=new password phrase addr specifies the password phrase that is to replace the user’s currently defined
password phrase. The address points to a 1-byte length field, followed by the password phrase, which can be 14-100 characters (or 9-100 characters if the new password phrase exit, ICHPWXX11, is installed and accepts the new value). Specifying a length field of zero is equivalent to not specifying NEWPHRASE.

RACF checks the following set of basic rules for the value specified by NEWPHRASE:

- The user ID is not part of the password phrase.
- At least two alphabetics are specified (A-Z, a-z).  
- At least two non-alphabetics are specified (numerics, punctuation, special characters, blanks).
- No more than two consecutive characters are identical.

If NEWPHRASE is specified without PHRASE, it is not used unless the user already has a password phrase, and PASSWRD is specified with a PassTicket instead of a password. If PASSWRD is specified with a PassTicket, and both NEWPASS and NEWPHRASE are specified, NEWPHRASE is used. A new password phrase cannot be set using a password for authentication, nor can a new password be set using a password phrase for authentication.

If NEWPHRASE is specified with PASSCHK=NO, no error message will be issued but the password phrase will not be changed.

When specifying NEWPHRASE=, you must also specify RELEASE=7730 or later.

\texttt{OIDCARD=oid addr}

specifies the address of the currently defined operator-identification card of the user who has entered the system. The address points to a 1-byte length field, followed by the operator ID card.

\texttt{PASSCHK=YES, PASSCHK=NO}

specifies whether the user's password, password phrase or OIDCARD is to be verified.

\texttt{YES}  
RACROUTE REQUEST=VERIFY verifies the user's password, password phrase or OIDCARD.

There are some circumstances where verification does not occur even though PASSCHK=YES is specified. Some examples are surrogate processing (see \textit{z/OS Security Server RACF Security Administrator's Guide}) or when the START or the ENVRIN keywords are specified.

\texttt{NO}  
The user's password, password phrase, or OIDCARD is not verified. And, if the logon is successful, no message is issued.

\texttt{PASSWRD=password addr}

specifies the currently defined password of the user who has entered the system. The address points to either:

- a 1-byte length field, followed by the password, which can be up to eight characters, or
- a 1-byte length field, followed by a PassTicket, which is always eight bytes.

\textbf{Note:} The currently defined password is maintained in the case entered, except when the following occurs: if the PASSASIS bit is off in the user's profile and the password does not match the current password in the user's
REQUEST=VERIFY

profile, the password is folded to uppercase and again compared to the current password provided MIXEDCASE PASSWORD support is enabled in SETROPTS.

,,PGMNAME=programmer name addr
   specifies the address of the name of the user who has entered the system. This 20-byte area is passed to the RACINIT installation exit routine; it is not used by the RACROUTE REQUEST=VERIFY routine.

,,PHRASE=password phrase addr
   specifies the currently defined password phrase of the user who has entered the system. The address points to a 1-byte length field, followed by the password phrase, which can be 9-100 characters. Specifying a length field of zero is equivalent to not specifying PHRASE.

   The PASSWRD and OIDCARD parameters will be not be used if the PHRASE parameter is specified.

   Password phrases will not be checked in cases where a password is not checked (PASSCHK=NO, START= or ENVRIN= specified, SURROGAT processing).

   When specifying PHRASE=, you must also specify RELEASE=7730 or later.

,,POE=port of entry addr
   specifies the address of the port of entry into the system. The address points to the name of the input device through which the user or job entered the system. For example, this could be the name of the input device through which the job was submitted or the terminal logged on to. The port of entry is an 8-character field that is left-justified and padded with blanks.

   The port of entry becomes a part of the user's security token (UTOKEN). A flag in the U TOKEN uniquely identifies the RACF general-resource class to which the data in the POE field belongs: APPCPORT, TERMINAL, CONSOLE, or JESINPUT. The SERVAUTH class can also be a port of entry but it must be specified using the SERVAUTH keyword.

   The RACF class, JESINPUT, provides the conditional access support for jobs entered into the system through a JES input device, and the CONSOLE class performs the same task for commands that originate from a console. The APPCPORT class provides conditional access support for users entering the system from a given LU (APPC port of entry).

   If the JESINPUT class is active and the JESINPUT profile protecting this port of entry has a security label other than SYSMULTI, it will override the user’s default security label if the SECLABEL keyword is not specified and the RACF option SECLBYSYSTEM is active on the system.

   The TERMINAL class covers the terminal used to log on to TSO.

   When both the POE and TER MID keywords, or both the POE and SERVAUTH keywords, are specified the POE keyword takes precedence. Information specified by POE= on an ENVIR=CREATE can be attached to the created ACEE and used in subsequent RACF processing. RACF does not make its own copy of this area when attaching this information to the created ACEE. This area must not be explicitly freed before the deletion of the ACEE. For the same reason, the area must reside in a non-task-related storage subpool so that implicit freeing of the area does not occur.

   Restriction: The POE keyword does not allow the length needed for a SERVAUTH resource representing an IP address.
REQUEST=VERIFY

,POENET=network name address
specifies the address of a structure that consists of a 1-byte length field followed by up to an 8-byte field containing the network name of the partner LU. When specified with the POE parameter, the value specified for POENET is combined with the value specified for POE to create a network qualified name in the form netid.luuname. The network qualified LU name is then used as the POE value during further processing. POENET is only valid with SESSION=APPCTP, and should not be specified with any other type of session. To specify the POENET parameter, you must specify RELEASE=2.6.

,REMOTE=YES
,REMOTE=NO
specifies whether the job came through the network. The default is REMOTE=NO.

,SECLABEL=seclabel addr
specifies the address of an 8-byte, left-justified character field containing the security label, padded to the right with blanks.

To use this keyword, you must specify RELEASE=1.9 or a later release number.

If you do not specify the SECLABEL parameter while the SECLABEL class is active, a security label may be derived from other parameters in the following order:
1. TOKNIN=
2. SERVAUTH=
3. TERMID=
4. JESINPUT (if SECLBYSYSTEM is active and the security label is other than SYSMULTI)
5. Default security label from user profile

If a security label was not found in any of these places, the user is assigned a security label of SYSLOW only when both of the following conditions are true:
• MLACTIVE is in effect.
• The user is authorized to the SYSLOW SECLABEL profile.

An installation can use security labels to establish an association between a specific RACF security level (SECLEVEL) and a set of (zero or more) RACF security categories (CATEGORY). If it is necessary to use security labels to prevent the unauthorized movement of data from one level to another when multiple levels of data are in use on the system at the same time, see z/OS Security Server RACF Security Administrator’s Guide for further information.

,SERVAUTH=serauth addr
specifies the address of the identifier for the server through which the user is accessing the system. The address points to a 1-byte length field followed by up to a 64-byte area containing the name of a resource in the SERVAUTH class. This resource is the network access security zone name that contains the IP address of the user. If the SERVAUTH class is active and the SERVAUTH profile protecting this resource has a security label other than SYSMULTI, it will override the user’s default security label if the SECLABEL keyword is not specified. After verifying that the user has access to this resource, a copy of the information specified by SERVAUTH= on an ENVIR=CREATE is attached to the created ACEE and used in subsequent RACF processing. If the POE keyword is specified, the SERVAUTH keyword is ignored. When the SERVAUTH keyword is specified, POE information in the STOKEN or TOKNIN and the TERMID keyword are ignored. When specifying SERVAUTH=, you must also specify RELEASE=7708 or later.
REQUEST=VERIFY

**Rule:** When both the POE and SERVAUTH parameters are specified, SERVAUTH is ignored.

,SESSION=type

specifies the session type to be associated with the request. Session types are literals. When the SESSION keyword is used in combination with the POE keyword, SESSION determines the class with which the POE keyword is connected.

When the session type is APPCTP, RACF requires APPL= and POE= also to be specified. The APPL= value should be the address of the local LU name, and the POE= value should be the address of the remote LU name.

If SERVAUTH is specified, the default session type is IP. If SERVAUTH is not specified and TERMID= or POE= is specified, the default session type is TSO. Otherwise, session type is not set.

**Restrictions for the IP session type:**

1. If a session type of IP is specified with the POE keyword, a parameter list abend will occur.
2. As with the OMVSSRV session type, the last access date and time messages are not issued.

The allowable session types and their associated POE classes are:

<table>
<thead>
<tr>
<th>Session type</th>
<th>Description</th>
<th>POE class</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPCTP</td>
<td>An APPC transaction program.</td>
<td>APPCPORT</td>
</tr>
<tr>
<td></td>
<td>When APPCTP is specified, user profile statistics are updated daily at most.</td>
<td></td>
</tr>
<tr>
<td>COMMAND</td>
<td>A command</td>
<td>CONSOLE</td>
</tr>
<tr>
<td>CONSOPER</td>
<td>A console operator</td>
<td>CONSOLE</td>
</tr>
<tr>
<td>EXTBATCH</td>
<td>A job from external reader (EXT)</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>EXTXBM</td>
<td>An execution batch monitor job</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>INTBATCH</td>
<td>A batch job from internal reader (INT)</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>INTXBM</td>
<td>An execution batch monitor job from INT</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>IP</td>
<td>A TCP/IP address</td>
<td>None</td>
</tr>
<tr>
<td>MOUNT</td>
<td>A mount</td>
<td>None</td>
</tr>
<tr>
<td>NJEBATCH</td>
<td>A job from network job entry (NJE)</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>NJEOPER</td>
<td>A network job entry operator</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>NJEXBM</td>
<td>A network execution batch monitor job</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>NJSYSOUT</td>
<td>A network SYSOUT</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>OMVSSRV</td>
<td>An OMVS server application</td>
<td>None</td>
</tr>
</tbody>
</table>

When OMVSSRV is specified, user profile statistics are updated daily at most. Audit records are only created when one of the following conditions are met:

- An incorrect password or password phrase is specified.
- The user ID has been revoked.
- A new password or password phrase was provided.
- A SECLABEL error occurred.
**REQUEST=VERIFY**

<table>
<thead>
<tr>
<th>Session type</th>
<th>Description</th>
<th>POE class</th>
</tr>
</thead>
<tbody>
<tr>
<td>RJEBATCH</td>
<td>A batch job from remote job entry (RJE)</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>RJEOPER</td>
<td>A remote job-entry operator</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>RJEXBM</td>
<td>A remote execution batch monitor job</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>STARTED</td>
<td>A started procedure of started task</td>
<td>None</td>
</tr>
<tr>
<td>SYSAS</td>
<td>A system address space</td>
<td>None</td>
</tr>
<tr>
<td>TKNUNKWN</td>
<td>An unknown user from NJE</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>TSO</td>
<td>A TSO or other interactive session logon</td>
<td>TERMINAL</td>
</tr>
</tbody>
</table>

**Note:** When there is no POE class associated with the session type, the POE ID and session are preserved.

,,SGROUP=submitting group addr

specifies the address of an area that contains a 1-byte length field followed by the group name of the user who submitted the unit of work. The group ID cannot exceed eight bytes.

,,SMC=YES
,,SMC=NO

specifies the use of the step-must-complete function of RACROUTE REQUEST=VERIFY processing.

YES  RACROUTE REQUEST=VERIFY processing makes other tasks for the job step non-dispatchable.

NO   The step-must-complete function is not used.

**Note:**
1. SMC=NO should not be used if DADSM ALLOCATE/SCRATCH functions execute simultaneously in the same address space as the RACROUTE REQUEST=VERIFY function.
2. When an automatic direction of application updates RACROUTE REQUEST=VERIFY request is issued with the SMC keyword specified, the SMC keyword is not propagated.

,,SNODE=submitting node addr

specifies the address of an area that contains a 1-byte length field followed by the name of the node from which the unit of work was submitted. The node name cannot exceed eight bytes.

,,START=procname addr

specifies the procedure name of the started task for which the RACROUTE REQUEST=VERIFY is being performed. The address points to an 8-byte area containing the procedure name (left-justified and padded with blanks, if necessary). If START= is specified, REQUEST=VERIFY processing searches the started procedure table for the user ID and group to use for this REQUEST=VERIFY request. If the USERID and GROUP keywords are specified, REQUEST=VERIFY uses those values if it cannot find a STARTED class profile or an entry in the started procedure table that matches the specified procedure name (and jobname from JOBNAME= if the STARTED class is used.)

If START is specified, PASSWRD and OIDCARD should not be specified.

,,STAT=ASIS
REQUEST=VERIFY

,STAT=NO
specifies whether the statistics controlled by the options specified on the RACF SETROPTS command should be maintained or ignored for this execution of RACROUTE REQUEST=VERIFY. This parameter also controls whether a message is to be issued when the logon is successful.

The default is ASIS.

Note: Messages are always issued if the RACROUTE REQUEST=VERIFY processing is unsuccessful.

ASIS
The messages and statistics are controlled by the installation's current options on the RACF SETROPTS command.

Note:
1. If SESSION=OMVSSRV or SESSION=APPCTP is specified, RACF updates the date and time of the user access at most once per day overriding the STAT=ASIS keyword.
2. For applications that specify the APPL operand, by specifying the RACF-INITSTATS(DAILY) string in the APPLDATA field, administrators can use the APPL class profiles, which are used to control which users can access the application, to specify which applications are to only have daily user statistics recorded. For more information, see the [z/OS Security Server RACF Security Administrator's Guide](https://www.ibm.com/support/knowledgecenter/en/SR298K_7.2.0/com.ibm.zos.security.racf.doc/), [Administrator's Guide](https://www.ibm.com/support/knowledgecenter/en/SR298K_7.2.0/com.ibm.zos.security.racf.doc/).

NO
The statistics are not updated. And, if the logon is successful, no message is issued.

When STA=NO is specified, the request does not result in the user being revoked even if the user's statistics have not been updated within k days, where k is the inactive period defined using SETROPTS INACTIVE(k).

,STOKEN=stoken addr

If you specify an STOKEN, the USERID in the STOKEN becomes the submitter's ID in the ACEE's token unless you specified the submitter's ID (SUSER) keyword. If you did, that keyword becomes the submitter's ID in the ACEE's token. Likewise, if you specified GROUP in STOKEN, that becomes the submitter's group in the ACEE's token, unless you specified the submitter's group (SGROUP) keyword. The SESSION, port-of-entry (POE), and port-of-entry class (POEX) fields are also used from the STOKEN. The execution node becomes the resulting submit node and execution node, unless you specify the submit node (SNODE) or execution-node address (EXENODE) keywords. In all cases, the specified keywords on the request override the fields of the STOKEN, if one is specified.

Also, STOKEN is used for surrogate checking, security-label dominance, or JESJOBS checking unless different submitter-checking information is supplied.

,SUBPOOL=subpool number
specifies the storage subpool from which the ACEE and related storage are
obtained. The value of the subpool can be literally specified or passed through a register. When using a register, the subpool number is the value of the least significant byte in the register.

If this parameter is not specified, it defaults to 255.

Note:
1. Take care in selecting a subpool, as MVS makes certain assumptions about subpool usage and characteristics. In particular, using subpool 0 or 250, or any subpool documented in \textit{z/OS MVS Programming: Assembler Services Guide} as having a storage key of \texttt{USER} (for example, 227-231, and 241) can give unpredictable results.

   In choosing a subpool, be aware that the storage obtained can be attached to MVS control blocks, so subpool characteristics need to be considered. Suppose, for example, that a task-related subpool is chosen for the ACEE; if you do not provide an anchor for the ACEE, in some cases the ACEE is attached to the MVS ASXB. When the task terminates, the ACEE storage is freed and the ASXB points to freemained storage.

   If a task related subpool is chosen, the application must ensure that only the task that created the ACEE is allowed to use it. Similarly, if a job step related subpool is chosen, only tasks that are associated with the same job step TCB as the task that created the ACEE may use that ACEE. Allowing other tasks (such as system or initiator tasks) to use it may cause unpredictable ABENDs.

2. If a common-area subpool (for example 226–228, 231, 239, 241, 245, 247, or 248) is used and not freed before the job terminates, then the job might show up in the exception reports of RMF (or other monitoring tools that support the tracking of common-area storage utilization) as owning common storage. Before your job terminates, it should issue a RACROUTE REQUEST=VERIFY, ENVIR=DELETE to free this common storage.

\texttt{SUSERID=submitting user id addr}

specifies the address of an area that contains a 1-byte length field, followed by the user ID of the user who submitted the unit of work. The user ID cannot exceed eight bytes.

Applications should fold the submitting user ID to uppercase.

\texttt{SYSTEM=NO}
\texttt{SYSTEM=YES}

specifies whether the caller is in supervisor state or system key 0–7, or both.

\texttt{NO} indicates that the caller cannot guarantee to be in supervisor state or system key 0–7.

\texttt{YES} indicates that the caller is in supervisor state and/or system key 0–7.

\texttt{SYSTEM=YES} is used to provide a fast path through RACROUTE REQUEST=VERIFY. However, unless \texttt{ENVRIN} is also specified, using any of the following keywords nullify the benefits of the fastpath:

\texttt{EXENO} \texttt{NESTED} \texttt{OIDCARD} \texttt{SGROUP} \texttt{STOKEN} \texttt{TOKNIN} \texttt{JOBNAME} \texttt{NEWPASS} \texttt{REMOTE} \texttt{SNODE} \texttt{SUSERID} \texttt{NEWPHRASE}

When \texttt{ENVRIN} is specified, the indicated keywords are ignored, therefore the “fastpath” benefit is recognized.

The default is \texttt{SYSTEM=NO}.

Note:
REQUEST=VERIFY

1. If the caller specifies SYSTEM=YES and is in neither supervisor state nor system key 0–7, the request abends.
2. Use of the SYSTEM=keyword requires that RELEASE=1.9.2 or later be specified.

,TERMID=terminal addr
specifies the address of the identifier for the terminal through which the user is accessing the system. The address points to an 8-byte area containing the terminal identifier. Information specified by TERMID= on an ENVIR=CREATE can be attached to the created ACEE and used in subsequent RACF processing. RACF does not make its own copy of this area when attaching this information to the created ACEE. This area must not be explicitly freed before the deletion of the ACEE. For the same reason, the area must reside in a non-task-related storage subpool so that implicit freeing of the area does not occur. If POE= is specified, the TERMID= area is not referred to in subsequent processing and can be freed at the user’s discretion.

If the TERMINAL class is active and the TERMINAL profile protecting this resource has a security label other than SYSMULTI, it will override the user’s default security label if the SECLABEL keyword is not specified.

Rule: When both the TERMID and SERVAUTH keywords are specified, the SERVAUTH keyword takes precedence.

,TOKNOUT=utoken addr
specifies an address that points to a user-provided area in which the UTOKEN is built. The mapping of the area is a 1-byte length field, followed by a 1-byte version code, followed by a 78-byte area in which to build the UTOKEN. This token is extracted from the ACEE built by this request.

,TOKNIN=utoken addr
specifies an address that points to a caller-provided area that contains an input UTOKEN. The mapping of the area is a 1-byte length field, followed by a 1-byte version code, followed by the UTOKEN itself, which can be 78 bytes long. The TOKNIN should have been previously obtained by RACROUTE REQUEST=VERIFY, VERIFYX, TOKENXTR or TOKENBLD.

,TRUSTED=YES, TRUSTED=NO
specifies whether the unit of work is a member of the trusted computer base. Subsequent RACROUTE REQUEST=AUTH requests using an ACEE with this attribute (or a token extracted from the ACEE) have the following effects:
- Authorization checking is bypassed (this includes bypassing the checks for security classification on users and data).
- No statistics are updated.
- No audit records are generated, except those requested using the SETROPTS LOGOPTIONS command or the UAUDIT operand on the ALTUSER command.
- No exits are called.

Subsequent RACROUTE REQUEST=FASTAUTH requests using an ACEE with this attribute (or a token extracted from the ACEE) have the following effects:
- Authorization checking is bypassed (this includes bypassing the checks for security classification on users and data).
- No statistics are updated.
- No audit records are generated, except those requested using the UAUDIT operand on the ALTUSER command.
**REQUEST=VERIFY**

This is similar to having the started procedures table trusted bit on.

**Note:** The TRUSTED=YES keyword only has meaning when SESSION=STARTED is also specified.

**USERID=userid addr**

specifies the user identification of the user who has entered the system. The address points to a 1-byte length field, followed by the user ID, which can be up to eight characters.

If the USERID= keyword is omitted, “*” is the default.

To prevent a protected user ID from being used to log on, RACROUTE REQUEST=VERIFY processing checks if the protected user ID indicator is on in the user profile. If the indicator is on, RACROUTE REQUEST=VERIFY fails unless keywords such as PASSCHK=NO or START=PROCNAME have been specified to indicate that no password is needed. If a password is expected to be specified for a RACROUTE, it fails as an incorrect password attempt. This denies users the ability to log on with a protected user ID using a password, PassTicket, or OID card. RACROUTE REQUEST=VERIFY processing returns the same RACROUTE return code, informational error message, and auditing as done for a normal system entry attempt with an incorrect password. However, the user profile is not updated, the revoke count is not incremented or reset, and the user is not revoked for exceeding the system limit for password attempts.

**Application considerations:** When verifying a user ID and password from a user, validate that the user ID and password are 1-8 characters in length and contain only characters that are alphabetic, numeric, # (X'7B'), @ (X'7C'), or $ (X'5B').

**Note:** If SETROPTS MIXEDCASE is not in effect, you must change the user ID, password, and new password to uppercase. If SETROPTS MIXEDCASE is in effect, you must change only the user ID to uppercase.

**Certificate user IDs:**

Certificate authority certificates are associated with the user ID irrcerta, MULTIID certificate name filters are associated with the user ID irrmulti, and site certificates are associated with the user ID irrsitec. These user IDs cannot be used for any purpose other than anchoring certificate authority certificates, site certificates, or certificate name filters.

The irrcerta, irrmulti, and irrsitec user IDs are defined to RACF during IPL in a manner similar to the method used to define the user ID IBMUSER. These user IDs are added in revoked status and are not connected to any groups, ensuring that they cannot be used as valid user IDs. RACROUTE REQUEST=VERIFYs performed for these user IDs fail due to the lack of connected groups.

**Identity context references:**

An 8-byte user ID with a prefix of “**” (X'5C5C') and an 8-byte password indicate that the USERID and PASSWRD parameters identify an identity context reference (ICR), not the user ID and password of a user that has entered the system. There is no change to the format of the user ID and password parameters.

When an identity context reference is specified on a RACROUTE REQUEST=VERIFY,ENVIR=CREATE request, and the SESSION=type is specified as (or defaulted to) IP, OMVSSRV, TSO, or no session type, RACF
attempts to resolve the reference from the local identity context cache using the R_cachserv callable service. It uses the identity context reference supplied as the USERID and PASSWRD parameters to determine the appropriate user ID, and other authentication information contained in an Identity Context Extension block (ICTX). If an ICR is specified with a different SESSION=type, RACF does not attempt to resolve it and the request proceeds with all of the parameters as specified.

When an ICR is specified for a valid session type, RACF does not utilize the following keywords in its subsequent processing: JOBNAME, SGROUP, SUSERID, SNODE, EXENODE, STOKEN, REMOTE, and START.

If the identity context reference is successfully resolved to a user ID and ICTX block, PASSCHK =NO will be set. PASSCHK=NO means that the following parameters will not be utilized during the authentication check: ENCRYPT, OIDCARD, PASSWRD, PHRASE, NEWPASS, or NEWPHRASE. RACF will continue authorization checking of the resolved user ID and, if successful, will build an ACEE that points to the ICTX block.

If the ICR could not be resolved, RACF will attempt to build an ACEE with the PASSCHK and USERID values as entered.

**X500NAME=X500 name pair addr**

specifies the data structure that contains the X.500 name pair associated with this security environment. Before using the name pair, you need to obtain it from the digital certificate associated with the user ID. You can use the initACEE query function for this task. The name pair must contain both the issuer’s name and the subject’s name from the certificate.

The X500NAME parameter is valid only for the ENVIR=CREATE function of a REQUEST=VERIFY request. However, the ENVIR=CREATE function ignores the parameter or uses a different name pair in certain circumstances:

- The parameter is ignored if both the ENVRIN parameter and SYSTEM=YES are specified.
- When creating an ACEE from an ENVR object, the ENVR object might already contain an X.500 name pair, which is used.
- If a RACROUTE REQUEST=VERIFY, ENVIR=CREATE creates an ACEE for an undefined user, the X500NAME parameter is ignored.
- If the RACROUTE REQUEST=VERIFY request creates an ACEE for a RACF defined user, the ACEE points to a copy of the X.500 name pair structure in the same subpool as the ACEE. This X.500 name is used in auditing.

When an A-type or RX-type notation is used, name pair addr specifies the field name of the data structure. When register notation is used, it specifies the register containing the address of the data structure.

When specifying X500NAME=, you must also specify RELEASE=7705 or later. The data structure of the X.500 name pair is shown in Table 14.

**Table 14. Description of X500NAME data structure**

<table>
<thead>
<tr>
<th>Offset</th>
<th>Length (bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4</td>
<td>Length of entire X.500 name pair data structure</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>Length of issuer’s name (1–255)</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>Length of subject’s name (1–255)</td>
</tr>
<tr>
<td>8</td>
<td>1–255</td>
<td>Up to 255 characters of the Issuer’s distinguished name, will be truncated if longer</td>
</tr>
</tbody>
</table>
Table 14. Description of X500NAME data structure  (continued)

<table>
<thead>
<tr>
<th>Offset</th>
<th>Length (bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>1–255</td>
<td>Up to 255 characters of the Subject’s distinguished name, will be truncated if longer</td>
</tr>
</tbody>
</table>

\textbf{,MF=S}

specifies the standard form for the RACROUTE REQUEST=VERIFY macro instruction.

**Guidelines for changing or deleting an ACEE**

Delete only an ACEE that you created. Issuing a RACROUTE REQUEST=VERIFY with ENVIR=DELETE specified to delete the existing ACEE can lead to problems if you were not the one who created that environment. The issuer of the ENVIR=CREATE that built the ACEE might have saved a pointer to it and might be expecting it to still be available later in processing. Note that this is the case for the initiator’s ACEE. Also, if you delete an ACEE, you might lose tables anchored off that ACEE that are needed later in RACF processing. See [z/OS Security Server RACF Diagnosis Guide](#) for overview diagrams of ACEEs and related control blocks. These diagrams can be useful when diagnosing problems.

**Note:** When you delete an ACEE that has a third-party ACEE attached, the RACINIT pre- or post-exits get control again for the third-party ACEE as well as for the original ACEE being deleted.

If you make a copy of the ACEE and update fields, avoid passing it to RACF. Many RACF services anchor tables off the ACEE and refresh these tables when required. If you update fields in a copy, the original ACEE contains incorrect pointers that result in abends when the original is used or deleted. In general, it is recommended that you do not copy an ACEE.

If you need to delete or change an ACEE that you did not create, you can use one of the following methods.

- **Save and restore the current ACEE pointers:**
  1. Save the current ACEE pointers from ASXBSENV and TCBSENV.
  2. Clear ASXBSENV and TCBSENV.
  3. Issue RACROUTE REQUEST=VERIFY ENVIR=CREATE.
  4. At the end of processing
     - a. Issue ENVIR=DELETE
     - b. Restore ASXBSENV and TCBSENV to the original values.

- **Change the values in the current ACEE:**
  - Issue RACROUTE REQUEST=VERIFY with ENVIR=CHANGE to change the values in the current ACEE.

- **Create, anchor, and delete a third-party ACEE:**
  - Issue RACROUTE REQUEST=AUTH with USERID= and GROUPID=, causing RACF to create, anchor, and delete a third-party ACEE internally.

**Return codes and reason codes**

When you execute the macro, space for the RACF return code and reason code is reserved in the first two words of the RACROUTE parameter list. You can access them using the ICHSAFP mapping macro, by loading the ICHSAFP pointer with the label that you specified on the list form of the macro. When control is returned,
**REQUEST=VERIFY**

Register 15 contains the SAF return code.

<table>
<thead>
<tr>
<th>SAF RC</th>
<th>Meaning</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACROUTE REQUEST=VERIFY has completed successfully.</td>
<td>00</td>
</tr>
<tr>
<td>04</td>
<td>Verify token information.</td>
<td>04</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0C</td>
<td>Indicates a TOKNIN was specified, but its length was too large.</td>
</tr>
<tr>
<td>10</td>
<td>Indicates an STOKEN was specified, but its length was too large.</td>
</tr>
</tbody>
</table>

| 04 | Requested function could not be completed. No RACF decision. |

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Indicates a normal completion.</td>
</tr>
</tbody>
</table>

| 04 | Verify token information. |

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACF was not called to process the request because one of the following occurred:</td>
</tr>
<tr>
<td></td>
<td>• ENVIR=VERIFY was specified without SAF installation exit processing.</td>
</tr>
<tr>
<td></td>
<td>• RACF is not installed.</td>
</tr>
<tr>
<td></td>
<td>• The combination of class, REQSTOR, and SUBSYS was found in the RACF router table, and ACTION=None was specified.</td>
</tr>
<tr>
<td></td>
<td>• The RACROUTE issuer specified DECOUPL=YES and a RELEASE= keyword with a higher release than is supported by this level of OS.</td>
</tr>
<tr>
<td>04</td>
<td>The user profile is not defined to RACF.</td>
</tr>
<tr>
<td>20</td>
<td>RACF is not active.</td>
</tr>
<tr>
<td>58</td>
<td>RJE or NJE operator FACILITY class profile not found.</td>
</tr>
</tbody>
</table>

| 08 | Requested function has failed. |

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>04</td>
<td>The user profile is not defined to RACF.</td>
</tr>
<tr>
<td>08</td>
<td>The password or password phrase is not authorized.</td>
</tr>
</tbody>
</table>

**Note:** All return and reason codes are shown in hexadecimal. Also, note that SAF return code is presented as SAF RC and RACF return code is presented as RACF RC in the following section.
REQUEST=VERIFY

0C The password or password phrase has expired.

10 At least one of the following conditions has occurred:
   - The new password or password phrase is not valid.
   - A new password phrase was specified with a current password,
     or a new password was specified with a current password
     phrase.
   - A new password phrase was specified with a PassTicket as the
     current password, but the user does not currently have a
     password phrase.
   - A password or password phrase change is disallowed at this
     time because the minimum password-change interval has not
     passed.

14 The user is not defined to the group.

18 RACROUTE REQUEST=VERIFY was failed by the installation exit
   routine.

1C The user's access has been revoked.

24 The user's access to the specified group has been revoked.

28 OIDCARD parameter is required but not supplied.

2C OIDCARD parameter is not valid for specified user.

30 The user is not authorized to the port of entry in the TERMINAL,
   JESINPUT, or CONSOLE class.

Reason Code
   Meaning

00 Indicates the user is not authorized to the port of entry.

04 Indicates the user is not authorized to access the system on
   this day, or at this time of day.

08 Indicates the port of entry cannot be used on this day, or at
   this time of day.

   Note: The port of entry refers to the TERMINAL class, the
   JESINPUT class, and the CONSOLE class ports of entry.

34 The user is not authorized to use the application.

38 SECLABEL checking failed.

Reason Code
   Meaning

04 MLACTIVE requires a SECLABEL; none was specified.

08 Indicates the user is not authorized to the SECLABEL.

0C The system was in a multilevel secure status, and the
   dominance check failed.

10 Neither the user's nor the submitter's security label
   dominates. They are disjoint.

14 The client's security label is not equivalent to the server's
   security label.

44 A default token is used as input token.
Indicates that an unprivileged user issued a RACROUTE REQUEST=VERIFY in a tranquil state (MLQUIET).

NODES checking failed.

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Submitter’s node is not allowed access to execution node.</td>
</tr>
<tr>
<td>04</td>
<td>NJE failure: UACC of NONE for USERID type of NODES profile.</td>
</tr>
<tr>
<td>08</td>
<td>NJE failure: UACC of NONE for GROUP type of NODES profile.</td>
</tr>
<tr>
<td>0C</td>
<td>NJE failure: UACC of NONE for SECLABEL type of NODES profile.</td>
</tr>
<tr>
<td>10</td>
<td>NJE failure: No local submit node specified.</td>
</tr>
<tr>
<td>14</td>
<td>NJE failure: Reverification of translated values failed.</td>
</tr>
</tbody>
</table>

Indicates that a surrogate submit attempt failed.

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>04</td>
<td>Indicates the SURROGAT class was inactive.</td>
</tr>
<tr>
<td>08</td>
<td>Indicates the submitter is not permitted by the user’s SURROGAT class profile.</td>
</tr>
<tr>
<td>0C</td>
<td>Indicates that the submitter is not authorized to the security label under which the job is to run.</td>
</tr>
</tbody>
</table>

Indicates that a JESJOBS check failed.

Indicates that an error occurred while retrieving data from the RACF database.

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0483yyyy</td>
<td>An error occurred while RACROUTE REQUEST=VERIFY was accessing the RACF data base. &quot;yyyy&quot; is the RACF manager return code associated with the abend that would have been issued.</td>
</tr>
</tbody>
</table>

Indicates that the CHECK subparameter of the RELEASE keyword was specified on the execute form of the RACROUTE REQUEST=VERIFY macro; however, the list form of the macro does not have the same release parameter. Macro processing terminates.

**Example 1**

Use the standard form of the macro to perform the following tasks:

- Create an ACEE for the user ID and its default group.
- Chain the ACEE off either the current TCB or ASXB, or both, by not specifying the ACEE keyword.
- Verify that the user named USERNAME is a valid user.
- Verify that the password called PASSWORD is valid.
REQUEST=VERIFY

RACROUTE REQUEST=VERIFY ENVIR=CREATE,USERID=USERNAME, PASSWRD=PASSWORD,WORKA=RACWK

Example 2
Use the standard form to perform the following tasks:
- Verify that the user named USERNAME is a valid user.
- Verify that the group named GROUPNAM is a valid group.
- Verify that USERNAME is defined to the group.
- Create an ACEE for the user and group and put its address in ACEEANCH.
- Specify that the user's password is not required.

RACROUTE REQUEST=VERIFY,ENVIR=CREATE,USERID=USERNAME, GROUP=GROUPNAM,ACEE=ACEEANCH, PASSCHK=NO,WORKA=RACWK

Example 3
Use the standard form of the macro to delete the ACEE of the current task or address space, or both.

RACROUTE REQUEST=VERIFY,ENVIR=DELETE,WORKA=RACWK

RACROUTE REQUEST=VERIFY (list form)

The list form of the RACROUTE REQUEST=VERIFY macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=VERIFY macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

| name | name: Symbol. Begin name in column 1. |
| b    | One or more blanks must precede RACROUTE. |

RACROUTE

b

One or more blanks must follow RACROUTE.

REQUEST=VERIFY

,ACEE=acee addr  acee addr: A-type address
,ACTINFO=account addr  account addr: A-type address
,APPL='applname'  applname: 1–8 character name
,APPL=applname addr  applname addr: A-type address
,ENCRYPT=YES Default: ENCRYPT=YES
,ENCRYPT=NO
REQUEST=VERIFY

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>,ENVIR=C</td>
<td>Default: ENVIR=CREATE</td>
</tr>
<tr>
<td>,ENVIR=V</td>
<td></td>
</tr>
<tr>
<td>,ENVIR=C</td>
<td></td>
</tr>
<tr>
<td>,ENVIR=D</td>
<td></td>
</tr>
<tr>
<td>,ENVIR=T</td>
<td></td>
</tr>
<tr>
<td>,ENVR=ENVR</td>
<td>envr data addr: A-type address</td>
</tr>
<tr>
<td>,ENVROUT=</td>
<td>envr data addr: A-type address</td>
</tr>
<tr>
<td>,ERROROPT=</td>
<td>Default: ERROROPT=ABEND</td>
</tr>
<tr>
<td>,ERROROPT=</td>
<td></td>
</tr>
<tr>
<td>,EXENSEN</td>
<td>execution node addr: A-type address</td>
</tr>
<tr>
<td>,GROUP=</td>
<td>group addr: A-type address</td>
</tr>
<tr>
<td>,ICRX=</td>
<td>icrx addr: A-type address</td>
</tr>
<tr>
<td>,ICTX=</td>
<td>ictx addr: A-type address</td>
</tr>
<tr>
<td>,IDID=</td>
<td>idid addr: A-type address</td>
</tr>
<tr>
<td>,INSTNL=</td>
<td>parm list addr: A-type address</td>
</tr>
<tr>
<td>,JOBNAME=</td>
<td>jobname addr: A-type address</td>
</tr>
<tr>
<td>,LOC=</td>
<td>Default: See parameter description.</td>
</tr>
<tr>
<td>,LOG=</td>
<td>Default: LOG=ASIS</td>
</tr>
<tr>
<td>,LOG=</td>
<td></td>
</tr>
<tr>
<td>,LOG=</td>
<td></td>
</tr>
<tr>
<td>,LOGSTR=</td>
<td>logstr addr: A-type address</td>
</tr>
<tr>
<td>,NESTED=</td>
<td>Default: NESTED=NO</td>
</tr>
<tr>
<td>,NESTED=</td>
<td></td>
</tr>
<tr>
<td>,NESTED=</td>
<td></td>
</tr>
<tr>
<td>,NESSAT=</td>
<td>new password addr: A-type address</td>
</tr>
<tr>
<td>,NEWPHRASE=</td>
<td>new password phrase addr: A-type address</td>
</tr>
<tr>
<td>,OIDCARD=</td>
<td>oid addr: A-type address</td>
</tr>
<tr>
<td>,PASSCHK=</td>
<td>Default: PASSCHK=YES</td>
</tr>
<tr>
<td>,PASSCHK=</td>
<td></td>
</tr>
<tr>
<td>,PASSCHK=</td>
<td></td>
</tr>
<tr>
<td>,PASSWRD=</td>
<td>password addr: A-type address</td>
</tr>
</tbody>
</table>
### REQUEST=VERIFY

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>.PGMNAME=programmer name addr</code></td>
<td>programmer name addr: A-type address</td>
</tr>
<tr>
<td><code>.PHRASE=password phrase addr</code></td>
<td>password phrase addr: A-type address</td>
</tr>
<tr>
<td><code>.POE=port of entry addr</code></td>
<td>port of entry addr: A-type address</td>
</tr>
<tr>
<td><code>.POENET=network name addr</code></td>
<td>network name addr: A-type address</td>
</tr>
<tr>
<td><code>.REMOTE=YES</code></td>
<td>Default: REMOTE=NO</td>
</tr>
<tr>
<td><code>.REMOTE=NO</code></td>
<td></td>
</tr>
<tr>
<td><code>.SECLABL=seclabel addr</code></td>
<td>seclabel addr: A-type address</td>
</tr>
<tr>
<td><code>.SERVAUTH=servauth addr</code></td>
<td>servauth addr: A-type address</td>
</tr>
</tbody>
</table>
| `.SESSION=type` | type: Any valid session type  
**Default:** SESSION=TSO |
| `.SGROUP=submitting group addr` | submitting group addr: A-type address |
| `.SMC=YES` | Default: SMC=YES |
| `.SMC=NO` |  |
| `.SNODE=submitting node addr` | submitting node addr: A-type address |
| `.START=procname addr` | procname addr: A-type address |
| `.STAT=ASIS` | Default: STAT=ASIS |
| `.STAT=NO` |  |
| `.STOKEN=token addr` | token addr: A-type address |
| `.SUBPOOL=subpool number` | subpool number: Decimal digit 0–255  
**Default:** See explanation of SUBPOOL keyword on page 227 |
| `.USERID=submitting userid addr` | submitting userid addr: A-type address |
| `.SYSTEM=NO` | Default: SYSTEM=NO |
| `.SYSTEM=YES` |  |
| **Note:** Use of the SYSTEM= keyword requires that RELEASE=1.9.2 or later be specified. |
| `.TERMID=terminal addr` | terminal addr: A-type address |
| `.TOKNIN=utoken addr` | utoken addr: A-type address |
| `.TOKNOUT=utoken addr` | utoken addr: A-type address |
| `.USERID=userid addr` | userid addr: A-type address |
| `.TRUSTED=YES` | Default: TRUSTED=NO |
| `.TRUSTED=NO` |  |
REQUEST=VERIFY

\X500NAME=X500 name
name pair addr: A-type address

\MF=L

The parameters are explained under the standard form of the RACROUTE REQUEST=VERIFY macro instruction with the following exception:

\MF=L
specifies the list form of the RACROUTE REQUEST=VERIFY macro instruction.

RACROUTE REQUEST=VERIFY (execute form)

The execute form of the RACROUTE REQUEST=VERIFY macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=VERIFY macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

name name: Symbol. Begin name in column 1.

RACROUTE

b
One or more blanks must precede RACROUTE.

RACROUTE

b
One or more blanks must follow RACROUTE.

REQUEST=VERIFY

,ACEE=acee addr
acee addr: Rx-type address or register (2) – (12)

,ACTINFO=account addr
account addr: Rx-type address or register (2) – (12)

,APPL=applname addr
applname addr: Rx-type address or register (2) – (12)

,ENCRYPT=YES
,ENCRYPT=NO

,ENVIR=CREATE
,ENVIR=VERIFY
,ENVIR=CHANGE
,ENVIR=DELETE

,ENVRIN=envr data addr
envr data addr: Rx-type address

,ENVROUT=envr data addr
envr data addr: Rx-type address

,ERROROPT=ABEND Default: ERROROPT=ABEND
,ERROROPT=NOABEND

,EXENODE=execution node addr
execution node addr: Rx-type address or register (2) – (12)

,GROUP=group addr
group addr: Rx-type address or register (2) – (12)
### REQUEST=VERIFY

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.ICRX=icrx addr</td>
<td>icrx addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.ICTX=ictx addr</td>
<td>ictx addr: Rx-type address or register (2) - (12)</td>
</tr>
<tr>
<td>.IDID=idid addr</td>
<td>idid addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.INSTLN=parm list addr</td>
<td>parm list addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.JOBNAME=jobname addr</td>
<td>jobname addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.LOC=BELOW</td>
<td></td>
</tr>
<tr>
<td>.LOC=ANY</td>
<td></td>
</tr>
<tr>
<td>.LOC=ABOVE</td>
<td></td>
</tr>
<tr>
<td><strong>Note:</strong> LOC can be coded if REQUEST=VERIFY or REQUEST=LIST is coded.</td>
<td></td>
</tr>
<tr>
<td>.LOG=ASIS</td>
<td></td>
</tr>
<tr>
<td>.LOG=ALL</td>
<td></td>
</tr>
<tr>
<td>.LOG=NULL</td>
<td></td>
</tr>
<tr>
<td>.LOGSTR=logstr addr</td>
<td>logstr addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.NESTED=YES</td>
<td></td>
</tr>
<tr>
<td>.NESTED=NO</td>
<td></td>
</tr>
<tr>
<td><strong>Default:</strong> NESTED=NO</td>
<td></td>
</tr>
<tr>
<td>.NESTED=COPY</td>
<td></td>
</tr>
<tr>
<td>.NEWPASS=new password addr</td>
<td>new password addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.NEWPHRASE=new password phrase addr</td>
<td>new password phrase addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.OIDCARD=oid addr</td>
<td>oid addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.PASSCHK=YES</td>
<td></td>
</tr>
<tr>
<td>.PASSCHK=NO</td>
<td></td>
</tr>
<tr>
<td>.PASSWD=password addr</td>
<td>password addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.PGMNAME=programmer name addr</td>
<td>programmer name addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.PHRASE=password phrase addr</td>
<td>password phrase addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.POE=port of entry addr</td>
<td>port of entry addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.POENET=network name addr</td>
<td>network name addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.RELEASE=number</td>
<td>number: See standard form</td>
</tr>
<tr>
<td>.RELEASE=(CHECK)</td>
<td></td>
</tr>
<tr>
<td><strong>Default:</strong> RELEASE=1.6</td>
<td></td>
</tr>
<tr>
<td>.RELEASE=(number,CHECK)</td>
<td></td>
</tr>
<tr>
<td>.REMOTE=YES</td>
<td></td>
</tr>
<tr>
<td>.REMOTE=NO</td>
<td></td>
</tr>
<tr>
<td>.SECLABL=seclabel addr</td>
<td>seclabel addr: Rx-type address or register (2) – (12)</td>
</tr>
</tbody>
</table>
### REQUEST=VERIFY

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SERVAUTH=servauth addr</td>
<td>servauth addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>SESSION=type</td>
<td>type: Any valid session type</td>
</tr>
<tr>
<td>SMC=YES</td>
<td></td>
</tr>
<tr>
<td>SMC=NO</td>
<td></td>
</tr>
<tr>
<td>SNODE=submitting node addr</td>
<td>submitting node addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>SGROUP=submitting group addr</td>
<td>submitting group addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>START=procname addr</td>
<td>procname addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>STAT=ASIS</td>
<td></td>
</tr>
<tr>
<td>STAT=NO</td>
<td></td>
</tr>
<tr>
<td>STOKEN=stoken addr</td>
<td>stoken addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>SUBPOOL=subpool number</td>
<td>subpool number: Decimal digit 0-255</td>
</tr>
<tr>
<td>SUBPOOL=reg</td>
<td>reg: Register (2) - (12)</td>
</tr>
<tr>
<td>SUBPOOL=reg</td>
<td></td>
</tr>
<tr>
<td>USERID=submitting userid addr</td>
<td>submitting userid addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>SYSTEM=NO</td>
<td></td>
</tr>
<tr>
<td>SYSTEM=YES</td>
<td></td>
</tr>
<tr>
<td>TERMID=terminal addr</td>
<td>terminal addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>TOKNIN=utoken addr</td>
<td>utoken addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>TOKNOUT=utoken addr</td>
<td>utoken addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>TRUSTED=YES</td>
<td></td>
</tr>
<tr>
<td>TRUSTED=NO</td>
<td></td>
</tr>
<tr>
<td>USERID=userid addr</td>
<td>userid addr: Rx-type address or register (2) - (12)</td>
</tr>
<tr>
<td>X500NAME=X500 name pair addr</td>
<td>name pair addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>MF=(E,ctrl addr)</td>
<td>ctrl addr: Rx-type address or register (1) or (2) – (12)</td>
</tr>
</tbody>
</table>

Note: Use of the SYSTEM= keyword requires that RELEASE=1.9.2 or later be specified.

The parameters are explained under the standard form of the RACROUTE REQUEST=VERIFY macro instruction with the following exceptions:

- **MF=(E,ctrl addr)** specifies the execute form of the RACROUTE REQUEST=VERIFY macro, using a remote, control-program parameter list.

- **RELEASE=number**
- **RELEASE=(,CHECK)**
REQUEST=VERIFY

\,RELEASE=(number, CHECK)

specifies the RACF release level of the parameter list to be generated by this macro.

To use the parameters associated with a release, you must specify the number of that release or a later release number. If you specify an earlier release level, the parameter is not accepted by macro processing, and an error message is issued at assembly time.

The default is RELEASE=1.6.

When you specify the RELEASE keyword, checking is done at assembly time. Compatibility between the list and execute forms of the RACROUTE REQUEST=VERIFY macro is validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.

RACROUTE REQUEST=VERIFY (modify form)

The modify form of the RACROUTE REQUEST=VERIFY macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=VERIFY macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
</tbody>
</table>

RACROUTE

| b    | One or more blanks must follow RACROUTE. |

REQUEST=VERIFY

\,ACEE=acee addr  acee addr: Rx-type address or register (2) – (12)

\,ACTINFO=account addr  account addr: Rx-type address or register (2) – (12)

\,APPL=applname addr  applname addr: Rx-type address or register (2) – (12)

\,ENCRYPT=YES, ENCRYPT=NO

\,ENVIR=CREATE, ENVIR=VERIFY, ENVIR=CHANGE, ENVIR=DELETE

\,ENVRIN=envr data addr  envr data addr: Rx-type address or register (2) – (12)

\,ENVROUT=envr data addr  envr data addr: Rx-type address or register (2) – (12)

\,ERROROPT=ABEND, ERROROPT=NOABEND  Default: ERROROPT=ABEND
REQUEST=VERIFY

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.EXENODE=exn</td>
<td>execution node addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.GROUP=gp</td>
<td>group addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.ICRX=icrx</td>
<td>icrx addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.ICTX=ictx</td>
<td>ictx addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.IDID=idd</td>
<td>idid addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.INSTLN=pl</td>
<td>parm list addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.JOBNAME=jn</td>
<td>jobname addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.LOC=BELOW</td>
<td></td>
</tr>
<tr>
<td>.LOC=ANY</td>
<td></td>
</tr>
<tr>
<td>.LOC=ABOVE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Note: LOC can be coded if REQUEST=VERIFY or REQUEST=LIST is coded.</td>
</tr>
<tr>
<td>.LOG=ASIS</td>
<td></td>
</tr>
<tr>
<td>.LOG=ALL</td>
<td></td>
</tr>
<tr>
<td>.LOG=NONE</td>
<td></td>
</tr>
<tr>
<td>.LOGSTR=logstr</td>
<td>logstr addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.NESTED=YES/NO</td>
<td>Default: NESTED=NO</td>
</tr>
<tr>
<td>.NEWPASS=pw</td>
<td>new password addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.NEWPHRASE=pp</td>
<td>new password phrase addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.OIDCARD=oid</td>
<td>oid addr: Rx-type address or register (2) – (12).</td>
</tr>
<tr>
<td>.PASSCHK=YES/NO</td>
<td></td>
</tr>
<tr>
<td>.PASSWORD=pword</td>
<td>password addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.PGMNAME=pname</td>
<td>programmer name addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.PHRASE=pp</td>
<td>password phrase addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.POE=pe</td>
<td>port of entry addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.POENET=en</td>
<td>network name addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>.REMOTE=YES/NO</td>
<td></td>
</tr>
</tbody>
</table>
REQUEST=VERIFY

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Example</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>SECLABEL=seclabel addr</em></td>
<td>Seclabel addr: Rx-type address or register (2) – (12)</td>
<td>seclabel addr</td>
<td></td>
</tr>
<tr>
<td><em>SERVAUTH=servauth addr</em></td>
<td>Servauth addr: Rx-type address or register (2) – (12)</td>
<td>servauth addr</td>
<td></td>
</tr>
<tr>
<td><em>SESSION=type</em></td>
<td>Type: Any valid session type</td>
<td>type</td>
<td></td>
</tr>
<tr>
<td><em>SGROUP=submitting group addr</em></td>
<td>Submitting group addr: Rx-type address or register (2) – (12)</td>
<td>submitting group addr</td>
<td></td>
</tr>
<tr>
<td><em>SMC=YES</em></td>
<td>SMC=NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>SNODE=submitting node addr</em></td>
<td>Submitting node addr: Rx-type address or register (2) – (12)</td>
<td>submitting node addr</td>
<td></td>
</tr>
<tr>
<td><em>START=procname addr</em></td>
<td>Procname addr: Rx-type address or register (2) – (12)</td>
<td>procname addr</td>
<td></td>
</tr>
<tr>
<td><em>STAT=ASIS</em></td>
<td>STAT=NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>STOKEN=token addr</em></td>
<td>Token addr: Rx-type address or register (2) – (12)</td>
<td>token addr</td>
<td></td>
</tr>
<tr>
<td><em>SUBPOOL=subpool number</em></td>
<td>Subpool number: Decimal digit 0-255</td>
<td>subpool number</td>
<td></td>
</tr>
<tr>
<td><em>SUSERID=submitting userid addr</em></td>
<td>Submitting userid addr: Rx-type address or register (2) – (12)</td>
<td>submitting userid addr</td>
<td></td>
</tr>
<tr>
<td><em>SYSTEM=NO</em></td>
<td>SYSTEM=YES</td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>TERMID=terminal addr</em></td>
<td>Terminal addr: Rx-type address or register (2) – (12)</td>
<td>terminal addr</td>
<td></td>
</tr>
<tr>
<td><em>TOKNIN=utoken addr</em></td>
<td>Utoken addr: Rx-type address or register (2) – (12)</td>
<td>utoken addr</td>
<td></td>
</tr>
<tr>
<td><em>TOKNOUT=utoken addr</em></td>
<td>Utoken addr: Rx-type address or register (2) – (12)</td>
<td>utoken addr</td>
<td></td>
</tr>
<tr>
<td><em>TRUSTED=YES</em></td>
<td>TRUSTED=NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>USERID=userid addr</em></td>
<td>Userid addr: Rx-type address or register (2) - (12)</td>
<td>userid addr</td>
<td></td>
</tr>
<tr>
<td><em>X500NAME=X500 name pair addr</em></td>
<td>Name pair addr: Rx-type address or register (2) – (12)</td>
<td>name pair addr</td>
<td></td>
</tr>
<tr>
<td><em>MF=(M,ctrl addr)</em></td>
<td>Ctrl addr: Rx-type address or register (1) or (2) – (12)</td>
<td>ctrl addr</td>
<td></td>
</tr>
</tbody>
</table>

The parameters are explained under the standard form of the RACROUTE REQUEST=VERIFY macro instruction with the following exception:

_\_{MF=(M,ctrl addr)}_  

specifies the modify form of the RACROUTE REQUEST=VERIFY macro, using a remote, control-program parameter list.
RACROUTE REQUEST=VERIFYX: Verify user and return a UTOKEN

The RACROUTE REQUEST=VERIFYX macro verifies a user and builds a UTOKEN based on the information passed in the parameter list, and handles the propagation of submitter ID.

If the caller specifies an already-existing STOKEN to VERIFYX, and if the caller additionally specifies any UTOKEN keywords on the request, the UTOKEN keywords that are specified override the corresponding parameters in the STOKEN that was passed. Thus, if the caller specifies an STOKEN, the caller should not specify any additional parameters unless the caller wants to supplement STOKEN information.

The following order of priority exists for replacing the fields in the existing TOKEN:

- Keywords specified on the request take precedence over corresponding fields in the TOKNIN and STOKEN parameters.
- All fields within the token specified by the TOKNIN keyword take precedence over those specified by STOKEN.
- The fields for the submitter’s ID, submitter’s group, submit node, execution node, session, port of entry and its class, as obtained from the token specified by the STOKEN keyword are last.

If you do not want certain fields overridden, do not specify keywords for those fields.

To use this service, you must specify RELEASE=1.9 or a later release number.

If RACF is not active or not installed, then SAF builds a default UTOKEN to satisfy the VERIFYX request. This action is indicated by the bit TOKDFLT being turned on in the mapped UTOKEN. If SAF cannot build a complete UTOKEN, it returns a UTOKEN containing all the information available from the RACROUTE parameter list. SAF returns the default UTOKEN at the address specified on the TOKNOUT keyword. If a valid security label was not specified on the call and could not be obtained from RACF, and if the system was in MLACTIVE fail, SAF returns a default security label of SYSHIGH if the caller specified TRUSTED=YES, and a value of SYSLOW if the caller specified TRUSTED=NO. SAF ensures the correct propagation of security information from a submitter to the unit of work if session type is INTBATCH or INTXBM. SAF also returns an error token (TOKERR) if verification fails.

To issue the RACROUTE REQUEST=VERIFYX macro, the calling module must be authorized (APF-authorized, in system key 0–7, or in supervisor state).

The caller cannot hold any locks when issuing RACROUTE REQUEST=VERIFYX.

Automatic direction of application updates does not propagate RACROUTE REQUEST=VERIFYX requests that update the RACF database; however, the following ICHEINTY requests issued by RACROUTE REQUEST=VERIFYX are propagated:

- The ICHEINTY setting the revoke flag in the user profile when a user is being revoked due to inactivity or incorrect password attempts.
- The ICHEINTY increasing the revoke count when a user enters an incorrect password.
REQUEST=VERIFYX

- The ICHEINTY that resets the revoke count to 0 when a user enters a valid password.

The ICHEINTY issued by RACROUTE REQUEST=VERIFYX to change the password in the user's profile is not eligible for propagation by automatic direction of application updates because it is already eligible for propagation by automatic password direction.

**Note:** Some functions (such as EXTRACT) require that an ACEE exist. RACF supports use of the *BYPASS* user ID to create an ACEE to satisfy these cases when no other ACEE is available.

To create this environment, specify the user ID as *BYPASS*, specify PASSCHK=NO, and do not specify a password field at all. A successful use is not audited but a failure may be audited if requested.

If an ACEE with a user ID of *BYPASS* is used with REQUEST=AUTH, a return code of 4 is returned. Depending upon the application, this may allow access to a resource.

RACROUTE REQUEST=VERIFYX (standard form)

The standard form of the RACROUTE REQUEST=VERIFYX macro is written as follows. For a description of additional keywords that you can code and additional parameters that are required on the RACROUTE request, but that are not specific to this request type, see "RACROUTE (standard form)" on page 13.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Symbol. Begin name in column 1.</td>
</tr>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
<tr>
<td>RACROUTE</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>One or more blanks must follow RACROUTE.</td>
</tr>
<tr>
<td>REQUEST=VERIFYX</td>
<td></td>
</tr>
<tr>
<td>,TOKNOUT=utoken addr</td>
<td>utoken addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,ACTINFO=account addr</td>
<td>account addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,APPL='applname'</td>
<td>applname: 1–8 character name</td>
</tr>
<tr>
<td>,APPL=applname addr</td>
<td>applname addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,ENCRYPT=YES</td>
<td>Default: ENCRYPT=YES</td>
</tr>
<tr>
<td>,ENCRYPT=NO</td>
<td></td>
</tr>
<tr>
<td>,ERROROPT=ABEND</td>
<td>Default: ERROROPT=ABEND</td>
</tr>
</tbody>
</table>
**REQUEST=VERIFYX**

<table>
<thead>
<tr>
<th>Macro</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>,ERROROPT=NOABEND</td>
<td></td>
</tr>
<tr>
<td>,EXENODE=execution node addr</td>
<td>execution node addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,GROUP=group addr</td>
<td>group addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,INSTLN=parm list addr</td>
<td>parm list addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,JOBNAME=jobname addr</td>
<td>jobname addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,LOG=ALL</td>
<td>Default: LOG=ASIS</td>
</tr>
<tr>
<td>,LOG=ASIS</td>
<td></td>
</tr>
<tr>
<td>,LOG=NONE</td>
<td></td>
</tr>
<tr>
<td>,LOGSTR=logstr addr</td>
<td>logstr addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,NEWPASS=new password addr</td>
<td>new password addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,NEWPHRASE=new password phrase addr</td>
<td>new password phrase addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,OIDCARD=oid addr</td>
<td>oid addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,PASSCHK=YES</td>
<td>Default: PASSCHK=YES</td>
</tr>
<tr>
<td>,PASSCHK=NO</td>
<td></td>
</tr>
<tr>
<td>,PASSWORD=password addr</td>
<td>password addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,PGMNAME=programmer name addr</td>
<td>programmer name addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,PHRASE=password phrase addr</td>
<td>password phrase addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,POE=port of entry addr</td>
<td>port of entry addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,POENET=network name addr</td>
<td>network name addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,REMOTE=YES</td>
<td>Default: REMOTE=NO</td>
</tr>
<tr>
<td>,REMOTE=NO</td>
<td></td>
</tr>
<tr>
<td>,SECLABL=seclabel addr</td>
<td>seclabel addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,SERVAUTH=servauth addr</td>
<td>servauth addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,SESSION=type</td>
<td>type: Any valid session type</td>
</tr>
<tr>
<td></td>
<td>Default: SESSION=TSO</td>
</tr>
<tr>
<td>,SGROUP=submitting group addr</td>
<td>submitting group addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,SNODE=submitting node addr</td>
<td>submitting node addr: A-type address or register (2) – (12)</td>
</tr>
</tbody>
</table>
**REQUEST=VERIFYX**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMC=YES</td>
<td></td>
<td>SMC=NO</td>
</tr>
<tr>
<td>procname addr</td>
<td>A-type address or register (2) – (12)</td>
<td></td>
</tr>
<tr>
<td>STAT=ASIS</td>
<td></td>
<td>STAT=NO</td>
</tr>
<tr>
<td>stoken addr</td>
<td>A-type address or register (2) – (12)</td>
<td></td>
</tr>
<tr>
<td>submitting userid addr</td>
<td>A-type address or register (2) – (12)</td>
<td></td>
</tr>
<tr>
<td>terminal addr</td>
<td>A-type address or register (2) – (12)</td>
<td></td>
</tr>
<tr>
<td>utoken addr</td>
<td>A-type address or register (2) – (12)</td>
<td></td>
</tr>
<tr>
<td>TRUSTED=YES</td>
<td></td>
<td>TRUSTED=NO</td>
</tr>
<tr>
<td>userid addr</td>
<td>A-type address or register (2) – (12)</td>
<td></td>
</tr>
<tr>
<td>MF=S</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The parameters are explained as follows:

- **ACTINFO=account addr**
  Specifies the address of a field containing accounting information. This 144-byte area is passed to the RACINIT installation exit routine; it is not used by the RACROUTE REQUEST=VERIFY routine. The accounting field, if supplied, should have the following format:
  - The first byte of the field contains the number (binary) of accounting fields.
  - The following bytes contain accounting fields, where each entry for an accounting field contains a 1-byte length field, followed by the field.

- **APPL=’applname’**
  Specifies the name of the application issuing the RACROUTE REQUEST=VERIFYX. If an address is specified, the address must point to an 8-byte application name, left justified and padded with blanks if necessary.

- **ENCRYPT=YES**
  Specifies whether RACROUTE REQUEST=VERIFYX encodes the old password, the new password, and the OIDCARD data passed to it.
  The default is YES.
    - **YES** Data specified by the PASSWRD, NEWPASS, and OIDCARD keywords are not pre-encoded. RACROUTE REQUEST=VERIFYX encodes the data before storing it in the user profile or using it to compare against stored data.
    - **NO** Data specified by the PASSWRD, NEWPASS, and OIDCARD keywords is already encoded. RACROUTE REQUEST=VERIFYX bypasses the encoding of this data before storing it in, or comparing it against, the user profile.
REQUEST=VERIFYX

Note: If the password was shipped from another system, the encryption method must be the same on all systems utilizing the password. For example, the RACF password authentication exit, ICHDEX01, must be identical on all systems.

ENCRIPT=NO does not apply to PHRASE and NEWPHRASE and will be ignored if specified.

,ERROROPT=ABEND
,ERROROPT=NOABEND

specifies whether RACROUTE REQUEST=VERIFYX will abend or not when an error occurs while it is accessing the RACF database.

The default is ABEND.

ABEND

When RACROUTE REQUEST=VERIFYX encounters an error accessing the RACF database, issue a 483 abend.

NOABEND

When RACROUTE REQUEST=VERIFYX encounters an error accessing the RACF database, 483 abends are suppressed. Instead, the request receives a SAF RC 8, RACF RC X'5C' and a RACF reason code of X'0483yyyy' where 'yyyy' is the RACF manager return code associated with the abend that would have been issued. If you are specifying the ERROROPT keyword with a specific release value, RELEASE=value, the Table 15 shows how the RELEASE= values affect the operation of the ERROROPT keyword:

Table 15. Relationship between the ERROROPT keyword and RELEASE= values

<table>
<thead>
<tr>
<th>Release</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>All earlier releases</td>
<td>ERROROPT keyword is flagged as an unknown keyword.</td>
</tr>
<tr>
<td>7703 and 7705</td>
<td>ERROROPT keyword is syntax checked only and an informational MNOTE indicating that the ERROROPT keyword is being ignored is returned. No abend suppression is performed. However, the SAF parameter list is built with the ERROROPT bit set. This allows programs which are assembled with RELEASE=7703 and RELEASE=7705 to take advantage of ERROROPT=NOABEND once the applications are executed in a z/OS Version 1 Release 3 (HBB7706) or later environment.</td>
</tr>
<tr>
<td>7706 and later</td>
<td>483 abends are replaced with a SAF return code of 8, a RACF return code of X'5C', and a RACF reason code of X'0483yyyy'. &quot;yyyy&quot; is the RACF manager return code associated with the abend that would have been issued.</td>
</tr>
</tbody>
</table>

,EXENODE=execution node addr

specifies the address of an area that contains a 1-byte length field followed by the name of the node on which the unit of work is to be executed. The node name cannot exceed eight bytes.

,GROUP=group addr

specifies the group of the user who has entered the system. The address points to a 1-byte length field, followed by the group name, which can be up to eight characters long.
REQUEST=VERIFYX

Applications should fold the group name to uppercase.

,INSTLN=parm list addr

specifies the address of an area containing parameter information meaningful
to the RACINIT installation exit routine. This area is passed to the installation
exit when the exit routine is given control from the RACROUTE
REQUEST=VERIFY routine.

The INSTLN parameter can be used by an installation having a user
verification or job initiation application, and wanting to pass information from
one installation module to the installation exit routine.

,JOBNAME=jobname addr

specifies the address of the job name of a background job. The address points
to an 8-byte area containing the job name (left-justified and padded with
blanks if necessary). If JOBNAME= is specified with the START= parameter,
and the STARTED class is active, RACF uses the jobname during its processing
to help determine the user ID and group name that are assigned for the started
task.

Note: The JOBNAME parameter is used by RACF during RACROUTE
REQUEST=VERIFYX authorization checking to verify the user's authority to
submit the job. It is also passed to the installation RACINIT exit routine.

,LOG=ALL
,LOG=ASIS
,LOG=NONE

specifies when log records are to be generated.

The default is LOG=ASIS.

ALL Any request to create an ACEE, regardless of whether it succeeds or
fails, generates a RACF log record.

ASIS Only those attempts to create an ACEE that fail generate RACF log
records.

NONE A request to create an ACEE, regardless of whether it succeeds or fails,
does not generate a RACF log record.

LOG=NONE suppresses both messages and SMF records regardless of
MSGSUPP=NO.

Note: SMF records are written for password changes when SETROPTS
AUDIT(USER) is in effect regardless of the LOG setting specified.

,LOGSTR=logstr addr

specifies the address of a 1-byte length field followed by character data that is
written to the SMF data set, together with RACF audit information.

,NEWPASS=new password addr

specifies the password that is to replace the user’s currently defined password.
The address points to a 1-byte length field, followed by the password, which
can be up to eight characters.

The NEWPASS= keyword has no effect unless PASSCHK=YES is either
defaulted to or explicitly specified and PASSWRD= is also specified. If the
NEWPASS= keyword is specified with PASSCHK=NO, no error message is
issued but the password is not changed. A new password phrase cannot be set
using a password for authentication, nor can a new password be set using a
password phrase for authentication.
REQUEST=VERIFYX

\NEWPHRASE\new password phrase addr

specifies the password phrase to replace the user's currently defined password phrase. The address points to a 1-byte length field, followed by the password phrase, which can be 14-100 characters (or 9-100 characters if the new password phrase exit, ICHPWXX11, is installed and accepts the new value). Specifying a length field of zero is equivalent to not specifying NEWPHRASE.

RACF checks the following set of basic rules for the value specified by NEWPHRASE:

- The user ID is not part of the password phrase.
- At least two alphabetics are specified (A - Z, a - z).
- At least two non-alphabetics are specified (numerics, punctuation, special characters, blanks).
- No more than two consecutive characters are identical.

If NEWPHRASE is specified without PHRASE, it is not used unless the user already has a password phrase, and PASSWRD is specified with a PassTicket instead of a password. If PASSWRD is specified with a PassTicket, and both NEWPASS and NEWPHRASE are specified, NEWPHRASE is used. A new password phrase cannot be set using a password for authentication, nor can a new password be set using a password phrase for authentication.

If NEWPHRASE is specified with PASSCHK=NO, no error message will be issued but the password phrase will not be changed.

When specifying NEWPHRASE=, you must also specify RELEASE=7730 or later.

\OIDCARD\oid addr

specifies the address of the currently defined operator-identification card of the user who has entered the system. The address points to a 1-byte length field, followed by the operator ID card.

\PASSCHK=YES
\PASSCHK=NO

specifies whether or not the user's password, password phrase or OIDCARD is to be verified.

YES  RACROUTE REQUEST=VERIFYX verifies the user's password, password phrase, or OIDCARD.

There are some circumstances where verification checking does not occur even though PASSCHK=YES is specified. Some examples are surrogate processing (see z/OS Security Server RACF Security Administrator's Guide) or when the START or the ENVRIN keywords are specified.

NO  The user's password or OIDCARD is not verified.

\PASSWRD\password addr

specifies the currently defined password of the user who has entered the system. The address points to either:

- a 1-byte length field, followed by the password, which can be up to eight characters, or
- a 1-byte length field, followed by a PassTicket, which is always eight bytes.

Note: The currently defined password is maintained in the case entered, except when the following occurs: if the PASSASIS bit is off in the user's profile and the password does not match the current password in the user's
REQUEST=VERIFYX

profile, the password is folded to uppercase and again compared to the current password provided. MIXEDCASE PASSWORD support is enabled in SETROPTS.

```
PGMNAME=program name addr
```
specifies the address of the name of the user who has entered the system. This 20-byte area is passed to the RACINIT installation exit routine; it is not used by RACF.

```
PHRASE=password phrase addr
```
specifies the address of the currently defined password phrase of the user who has entered the system. The address points to a 1-byte length field followed by the password phrase, which can be 9-100 characters. Specifying a length field of zero is equivalent to not specifying PHRASE.

The PASSWRD and OIDCARD parameters are not used if the PHRASE parameter is specified.

Password phrases are not checked in cases where a password is not checked (PASSCHK=NO, START= or ENVIRIN= specified, SURROGAT processing).

When specifying PHRASE=, you must also specify RELEASE=7730 or later.

```
POE=port of entry addr
```
specifies the address of the port of entry into the system. The address points to the name of the input device through which the user or job entered the system. For example, this could be the name of the input device through which the job was submitted or of the terminal logged onto. The port of entry is an 8-character field that is left-justified and padded with blanks.

The port of entry becomes a part of the user’s security token (UTOKEN). A flag in the UTOKEN uniquely identifies the RACF general-resource class to which the data in the POE field belongs: APPCPORT, TERMINAL, CONSOLE, or JESINPUT. The SERVAUTH class can also be a port of entry but it must be specified using the SERVAUTH keyword.

The RACF class JESINPUT provides the conditional access support for jobs entered into the system through a JES input device. The CONSOLE class performs the same task for commands that originate from a console. In addition, the APPCPORT class provides conditional access support for users entering the system from a given LU (APPC port of entry).

If the JESINPUT class is active and the JESINPUT profile protecting this port of entry has a security label other than SYSMULTI, it will override the user’s default security label if the SECLABEL keyword is not specified and the RACF option SECLBYSYSTEM is active on the system.

The TERMINAL class covers the terminal used to log onto TSO.

When both the POE and TERMID keywords, or both the POE and SERVAUTH keywords, are specified the POE keyword takes precedence. Information specified by POE= on an ENVIR=CREATE can be attached to the created ACEE and used in subsequent RACF processing. RACF does not make its own copy of this area when attaching this information to the created ACEE. This area must not be explicitly freed prior to the deletion of the ACEE. For the same reason, the area must reside in a non-task-related storage subpool so that implicit freeing of the area does not occur.

**Restriction:** The POE keyword does not allow the length needed for a SERVAUTH resource representing an IP address.
\textbf{REQUEST=VERIFYX}

\textbf{POENET=network name address}

specifies the address of a structure that consists of a 1-byte length field followed by up to an 8-byte field containing the network name of the partner LU. When specified with the POE parameter, the value specified for POENET is combined with the value specified for POE to create a network qualified name in the form \textit{netid.luname}. The network qualified LU name is then used as the POE value during further processing. POENET is only valid with \texttt{SESSION=APPCTP}, and should not be specified with any other type of session. To specify the POENET parameter, you must specify \texttt{RELEASE=2.6}.

\textbf{REMOTE=YES, REMOTE=NO}

specifies whether or not the job came through the network. The default is \texttt{REMOTE=NO}.

\textbf{SECLABEL=seclabel addr}

specifies the address of an 8-byte, left-justified character field containing the security label, padded to the right with blanks.

If you do not specify the SECLABEL parameter while the SECLABEL class is active, a security label may be derived from other parameters in the following order:

1. \texttt{TOKNIN=}
2. \texttt{SERVAUTH=}
3. \texttt{TERMID=}
4. \texttt{JESINPUT} (if SECLBYSYSTEM is active and the security label is other than SYSMULTI)
5. Default security label from user profile

If a security label was not found in any of these places, the user is assigned a security label of SYSLOW only when both of the following conditions are true:

- MLACTIVE is in effect.
- The user is authorized to the SYSLOW SECLABEL profile.

An installation can use security labels to establish an association between a specific RACF security level (SECLEVEL) and a set of (zero or more) RACF security categories (CATEGORY). If it is necessary to use security labels to prevent the unauthorized movement of data from one level to another when multiple levels of data are in use on the system at the same time, see \textit{z/OS Security Server RACF Security Administrator’s Guide} for further information.

\textbf{SERVAUTH=servauth addr}

specifies the address of the identifier for the server through which the user is accessing the system. The address points to a 1-byte length field followed by up to a 64-byte area containing the name of a resource in the SERVAUTH class. This resource is the network access security zone name that contains the IP address of the user. If the SERVAUTH class is active and the SERVAUTH profile protecting this resource has a security label other than SYSMULTI, it will override the user's default security label if the SECLABEL keyword is not specified. After verifying that the user has access to this resource, a copy of the information specified by SERVAUTH= on an \texttt{ENVIR=CREATE} is attached to the created ACEE and used in subsequent RACF processing. If the POE keyword is specified, the SERVAUTH keyword is ignored. When the SERVAUTH keyword is specified, POE information in the STOKEN or TOKNIN and the TERMID keyword are ignored. When specifying SERVAUTH=, you must also specify \texttt{RELEASE=7708} or later.
Rule: When both the POE and SERVAUTH parameters are specified, SERVAUTH is ignored.

\$SESSION=type

specifies the session type to be associated with the request. Session types are literals. When the SESSION keyword is used in combination with the POE keyword, SESSION determines the class with which the POE keyword is connected.

When the session type is APPCTP, RACF requires APPL= and POE= also to be specified. The APPL= value should be the address of the local LU name, and the POE= value should be the address of the remote LU name.

If SERVAUTH is specified, the default session type is IP. If SERVAUTH is not specified and TERMID= or POE= is specified, the default session type is TSO. Otherwise, session type is not set.

Restrictions for the IP session type:

1. If a session type of IP is specified with the POE keyword, a parameter list abend will occur.
2. As with the OMVSSRV session type, the last access date and time messages are not issued.

The allowable session types and their associated POE classes are:

<table>
<thead>
<tr>
<th>Session type</th>
<th>Description</th>
<th>POE class</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPCTP</td>
<td>An APPC transaction program</td>
<td>APPCPORT</td>
</tr>
<tr>
<td>COMMAND</td>
<td>A command</td>
<td>CONSOLE</td>
</tr>
<tr>
<td>CONSOPER</td>
<td>A console operator</td>
<td>CONSOLE</td>
</tr>
<tr>
<td>EXTBATCH</td>
<td>A job from external reader (EXT)</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>EXTXBM</td>
<td>An execution batch monitor job</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>INTBATCH</td>
<td>A batch job from internal reader (INT)</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>INTXBM</td>
<td>An execution batch monitor job from INT</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>IP</td>
<td>A TCP/IP address</td>
<td>None</td>
</tr>
<tr>
<td>MOUNT</td>
<td>A mount</td>
<td>None</td>
</tr>
<tr>
<td>NJEBATCH</td>
<td>A job from network job entry (NJE)</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>NJEOPER</td>
<td>A network job-entry operator</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>NJEXBM</td>
<td>An network execution batch monitor job</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>NJSYSOUT</td>
<td>A network SYSOUT</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>OMVSSRV</td>
<td>An OMVS server application</td>
<td>None</td>
</tr>
<tr>
<td>RJEBATCH</td>
<td>A batch job from remote job entry (RJE)</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>RJEOPER</td>
<td>A remote job-entry operator</td>
<td>JESINPUT</td>
</tr>
</tbody>
</table>

When OMVSSRV is specified, user profile statistics are updated daily at most. Audit records are only created when one of the following conditions are met:

- An incorrect password or password phrase is specified.
- The user ID has been revoked.
- A new password or password phrase was provided.
- A security label error occurred.
REQUEST=VERIFYX

<table>
<thead>
<tr>
<th>Session type</th>
<th>Description</th>
<th>POE class</th>
</tr>
</thead>
<tbody>
<tr>
<td>RJEXBM</td>
<td>A remote execution batch monitor job</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>STARTED</td>
<td>A started procedure of started task</td>
<td>None</td>
</tr>
<tr>
<td>SYSAS</td>
<td>A system address space</td>
<td>None</td>
</tr>
<tr>
<td>TKNUNKWN</td>
<td>An unknown user from NJE</td>
<td>JESINPUT</td>
</tr>
<tr>
<td>TSO</td>
<td>A TSO or other interactive session logon</td>
<td>TERMINAL</td>
</tr>
</tbody>
</table>

**Note:** When no POE class is associated with the session type, the POE ID and session are preserved.

,SGROUP=submitting group addr
specifies the address of an area that contains a 1-byte length field followed by the group name of the user who submitted the unit of work. The group name cannot exceed eight bytes.

,SMC=YES
,SMC=NO
specifies the use of the step-must-complete function of RACROUTE REQUEST=VERIFYX processing.

YES  RACROUTE REQUEST=VERIFYX processing makes other tasks for the step non-dispatchable.
NO   The step-must-complete function is not used.

**Note:** SMC=NO should not be used if DADSM ALLOCATE/SCRATCH functions execute simultaneously in the same address space as the RACROUTE REQUEST=VERIFYX function.

,SNODE=submitting node addr
specifies the address of an area that contains a 1-byte length field, followed by the name of the node from which the unit of work was submitted. The node name cannot exceed eight bytes.

,START=procname addr
specifies the procedure name of a started task for which the RACROUTE REQUEST=VERIFYX is being performed. The address points to an 8-byte area containing the procedure name (left-justified and padded with blanks if necessary). If START= is specified, REQUEST=VERIFYX processing searches the started-procedures table for the user ID and group to use for this REQUEST=VERIFYX request. If the USERID and GROUP keywords are specified, REQUEST=VERIFYX uses those values if it cannot find a STARTED class profile or an entry in the started procedure table that matches the specified procedure name (and jobname from JOBNAME= if the STARTED class is used.)

If START is specified, PASSWRD and OIDCARD should not be specified.

,STAT=ASIS
,STAT=NO
specifies that no statistics are updated for this execution of RACROUTE REQUEST=VERIFYX, and that if logon is successful, no message is issued.

When STAT=NO is specified, the request does not result in the user being revoked even if the user’s statistics have not been updated within \( k \) days (where \( k \) is the inactive period defined using SETROPTS INACTIVE(\( k \))).

**Note:**
REQUEST=VERIFYX

1. The default (STAT=ASIS) is processed the same as STAT=NO.
2. Messages are always issued if the RACROUTE REQUEST=VERIFYX processing is unsuccessful.

\texttt{STOKEN=stoken addr}

specifies the address of the submitter's security token (UTOKEN). The first byte contains the length of the UTOKEN, and the second byte contains the format version number. See ICHRUTKN mapping. See “RUTKN: Resource/User Security Token” in z/OS Security Server RACF Data Areas.

If you specify STOKEN, the user ID in STOKEN becomes the submitter's ID in TOKNOUT, unless you specified the submitter's ID (SUSER) keyword. If you did, that keyword becomes the submitter's ID in TOKNOUT. Likewise, if you specified GROUP in the STOKEN, that becomes the submitter's group in TOKNOUT, unless you specified the submitter's group (SGROUP) keyword. The SESSION, port-of-entry (POE), and port-of-entry class (POEX) fields are also used from the STOKEN. The execution node becomes the resulting submit node and execution node unless you specify the submit node (SNODE) or execution node (EXENODE) keywords. In all cases, the specified keywords on the request override the fields of the STOKEN, if one is specified.

Also, STOKEN is used unless different submitter-checking information, such as surrogate checking, security-label dominance, or JESJOBS checking is specified.

\texttt{SUSERID=submitting userid addr}

specifies the address of an area that contains a 1-byte length field followed by the user ID of the user who submitted the unit of work. The user ID cannot exceed eight bytes.

Applications should fold the submitting user ID to uppercase.

\texttt{TERMID=terminal addr}

specifies the address of the identifier for the terminal through which the user is accessing the system. The address points to an 8-byte area containing the terminal identifier. The area must reside in a storage subpool not related to any task.

If POE= is specified, the TERMID= area is not referred to in subsequent processing and can be freed at the user's discretion. If the TERMINAL class is active and the TERMINAL profile protecting this resource has a security label other than SYSMULTI, it will override the user's default security label if the SECLABEL keyword is not specified.

Rule: When both the TERMID and SERVAUTH keywords are specified, the SERVAUTH keyword takes precedence.

\texttt{TOKNIN=utoken addr}

specifies an address that points to a caller-provided area that contains an input UTOKEN. The mapping of the area is a 1-byte length field, followed by a 1-byte version code, followed by the UTOKEN itself, which can be 78 bytes long. The TOKNIN should have been previously obtained by RACROUTE REQUEST=VERIFY, VERIFYX, TOKENXTR or TOKENBLD.

\texttt{TOKNOUT=output token addr}

specifies the address of the caller-provided area in which the UTOKEN is built. The first byte of storage at the address specified is the token length field. The second byte must contain the format version of the token. It is followed by a 78-byte area in which to build the UTOKEN. The mapping of the area is a 1-byte length field, followed by a 1-byte version code, followed by the rest of the token information.
REQUEST=VERIFYX

For a description of the fields TOKNOUT uses from STOKEN, see the STOKEN description.

TRUSTED=YES, TRUSTED=NO
specifies whether or not the unit of work is a member of the trusted computer base. Subsequent RACROUTE REQUEST=AUTH requests using a token with this attribute have the following effects:
- Authorization checking is bypassed (this includes bypassing the checks for security classification on users and data).
- No statistics are updated.
- No audit records are generated, except those requested using the SETROPTS LOGOPTIONS command or the UAUDIT operand on the ALTUSER command.
- No exits are called.

Subsequent RACROUTE REQUEST=FASTAUTH requests using a token with this attribute have the following effects:
- Authorization checking is bypassed (this includes bypassing the checks for security classification on users and data).
- No statistics are updated.
- No audit records are generated, except those requested using the UAUDIT operand on the ALTUSER command.

This is similar to having the started-procedures-table trusted bit on.

Note: The TRUSTED=YES keyword only has meaning when SESSION=STARTED is also specified.

USERID=userid addr
specifies the user identification of the user who has entered the system. The address points to a 1-byte length field, followed by the user ID, which can be up to eight characters.

If the USERID= keyword is omitted, (*) is the default.

To prevent a protected user ID from being used to logon, RACROUTE REQUEST=VERIFYX processing checks for the protected user ID being specified, and fails for requests that have a password specified or expected. For additional information on RACROUTEs handling of protected user IDs, see the USERID parameter of RACROUTE REQUEST=VERIFY.

Application considerations: When verifying a user ID and password from a user, be sure to validate that the user ID and password contain only alphanumeric characters and are 1–8 characters in length. Additionally, you must change the user ID, password, and new password to uppercase unless SETROPTS MIXEDCASE is in effect. If SETROPTS MIXEDCASE is in effect, you must change only the user ID to uppercase.

Certificate user IDs:
Certificate authority certificates are associated with the user ID irrcerta, MULTIID certificate name filters are associated with the user ID irmulti, and site certificates are associated with the user ID irrsitec. These user IDs cannot be used for any purpose other than anchoring certificate authority certificates, site certificates, or certificate name filters.

The irrcerta, irmulti, and irrsitec user IDs are defined to RACF during IPL in a manner similar to the method used to define the user ID IBMUSER.
REQUEST=VERIFYX

These user IDs are added in revoked status and are not connected to any groups, ensuring that they cannot be used as valid user IDs. RACROUTE REQUEST=VERIFYs performed for these user IDs fail due to the lack of connected groups.

\texttt{,MF=S}

specifies the standard form of the RACROUTE REQUEST=VERIFYX macro instruction.

Return codes and reason codes

When you execute the macro, space for the RACF return code and reason code is reserved in the first two words of the RACROUTE parameter list. You can access them using the ICHSAFP mapping macro, by loading the ICHSAFP pointer with the label that you specified on the list form of the macro. When control is returned, register 15 contains the SAF return code.

<table>
<thead>
<tr>
<th>SAF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACROUTE REQUEST=VERIFYX has completed successfully.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RACF RC</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>3C</td>
<td>Request completed successfully, but a VERIFYX condition occurred in SAF.</td>
</tr>
</tbody>
</table>

**Reason Code**

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
</tr>
<tr>
<td>24</td>
</tr>
<tr>
<td>30</td>
</tr>
</tbody>
</table>

| 04     | The requested function could not be performed. |

**Reason Code**

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>00</th>
<th>RACF was not called to process the request because one of the following occurred:</th>
</tr>
</thead>
<tbody>
<tr>
<td>•</td>
<td>RACF is not installed.</td>
</tr>
<tr>
<td>•</td>
<td>The combination of class, REQSTOR, and SUBSYS was found in the RACF router table, and ACTION=NONE was specified.</td>
</tr>
<tr>
<td>•</td>
<td>The RACROUTE issuer specified DECOUPL=YES and a RELEASE= keyword with a higher release than is supported by this level of z/OS.</td>
</tr>
</tbody>
</table>
REQUEST=VERIFYX

20 RACF is not active.
3C RACF is not installed.
58 RJE or NJE operator FACILITY class profile not found.

08 The requested function failed.

RACF RC
  Meaning
  00 Default ACEE or token-building error.

Reason Code
  Meaning
  00 SAF failed to set up a recovery environment.

  04 The user profile is not defined to RACF.

  08 The password or password phrase is not authorized.

  0C The password or password phrase has expired.

  10 At least one of the following conditions has occurred:
    • The new password or password phrase is not valid.
    • A new password phrase was specified with a current password,
      or a new password was specified with a current password
      phrase.
    • A new password phrase was specified with a PassTicket as the
      current password, but the user does not currently have a
      password phrase.
    • A password or password phrase change is disallowed at this
      time because the minimum password-change interval has not
      passed.

  14 The user is not defined to the group.

  18 RACROUTE REQUEST=VERIFYX was failed by the installation
      exit routine.

  1C The user's access has been revoked.

  24 The user's access to the specified group has been revoked.

  28 OIDCARD parameter is required but not supplied.

  2C OIDCARD parameter is not valid for specified user.

  30 The user is not authorized to the port of entry.

  34 The user is not authorized to use the application.

  38 SECLABEL checking failed.

Reason Code
  Meaning
  04 MLACTIVE requires a security label; none was specified.

  08 Indicates the user is not authorized to the security label.

  0C The system was in multilevel secure status, and the
      dominance check failed.

  10 Neither the user's nor the submitter's security label
      dominates. They are disjoint.
### REQUEST=VERIFYX

<table>
<thead>
<tr>
<th>Code</th>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td></td>
<td>The client's security label is not equivalent to the server's security label.</td>
</tr>
<tr>
<td>3C</td>
<td></td>
<td>A VERIFYX error occurred in SAF.</td>
</tr>
</tbody>
</table>

#### Reason Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>04</td>
<td>Old password required. Message IRR009I issued.</td>
</tr>
<tr>
<td>08</td>
<td>User ID required. Message IRR008I issued.</td>
</tr>
<tr>
<td>0C</td>
<td>Propagation checking could not complete. Failed to set up a recovery environment.</td>
</tr>
<tr>
<td>44</td>
<td>A default token is used as input token.</td>
</tr>
<tr>
<td>48</td>
<td>Indicates that an unprivileged user issued a RACROUTE REQUEST=VERIFYX in a tranquil state (MLQUIET).</td>
</tr>
<tr>
<td>4C</td>
<td>NODES checking failed.</td>
</tr>
</tbody>
</table>

#### Reason Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Submitter's node is not allowed access to execution node.</td>
</tr>
<tr>
<td>04</td>
<td>NJE failure: UACC of NONE for USERID type of NODES profile.</td>
</tr>
<tr>
<td>08</td>
<td>NJE failure: UACC of NONE for GROUP type of NODES profile.</td>
</tr>
<tr>
<td>0C</td>
<td>NJE failure: UACC of NONE for SECLABEL type of NODES profile.</td>
</tr>
<tr>
<td>10</td>
<td>NJE failure: No local submit node specified.</td>
</tr>
<tr>
<td>14</td>
<td>NJE failure: Reverification of translated values failed.</td>
</tr>
<tr>
<td>50</td>
<td>Indicates that a surrogate submit attempt failed.</td>
</tr>
</tbody>
</table>

#### Reason Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>04</td>
<td>Indicates the SURROGAT class was inactive.</td>
</tr>
<tr>
<td>08</td>
<td>Indicates the submitter is not permitted by the user's SURROGAT class profile.</td>
</tr>
<tr>
<td>0C</td>
<td>Indicates that the submitter is not authorized to the security label under which the job is to run.</td>
</tr>
<tr>
<td>54</td>
<td>Indicates that a JESJOBS check failed.</td>
</tr>
<tr>
<td>5C</td>
<td>Indicates that an error occurred while retrieving data from the RACF database.</td>
</tr>
</tbody>
</table>

#### Reason Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0483yyyy</td>
<td>An error occurred while RACROUTE REQUEST=VERIFY was accessing the RACF data base. &quot;yyyy&quot; is the RACF manager return code associated with the abend that would have been issued.</td>
</tr>
<tr>
<td>64</td>
<td>Indicates that the CHECK subparameter of the RELEASE keyword</td>
</tr>
</tbody>
</table>
was specified on the execute form of the RACROUTE REQUEST=VERIFYX macro; however, the list form of the macro does not have the same release parameter. Macro processing terminates.

**Example 1**

The following example shows a RACROUTE REQUEST=VERIFYX coded for handling verification checking for a batch job that has been submitted with a USERID, GROUPID, SECLABEL, and PASSWORD. The U_TOKEN area is filled with the verified job information.

```
RACROUTE REQUEST=VERIFYX,SESSION=INTBATCH, X
    PASSWRD=PASSWORD, TOKNOUT=TOKOUT, X
    EXENODE=EXNOD, USERID=USER, X
    GROUP=GROUPID, SECLABL=SLBL, X
    STOKEN=STOK, TRUSTED=NO, WORKA=RACWK, X
    RELEASE=1.9
```

```
:  
  PASSWORD DS OCL9
  PASSWL DS FLI'5'
  PASSWT DS CL8'PWD01'
  TOKOUT DS OCL80
  TKOLEN DS XL1'50' /* LENGTH - 80 DEC */
  TKOVRS DS XL1'01' /* VERSION 1 */
  TKODATA DS CL78
  EXNOD DS OCL9
  EXNODL DS FLI'2'
  EXNODT DS CL8'SYS1'
  USER DS OCL9
  USERL DS FLI'6'
  USERT DS CL8'USER01'
  GROUPID DS OCL9
  GROUPIDL DS FLI'4'
  GROUPIDT DS CL8'SYS1'
  SLBL DS CL8'SYSLOW'
  STOK DS CL80 /* OBTAINED BY PREVIOUS RACROUTE CALL */
  RACWK DS CL512
```

**Note:** Additional keywords required by RACF to complete the request, such as WORKA, are specified on RACROUTE itself.

---

**RACROUTE REQUEST=VERIFYX (list form)**

The list form of the RACROUTE REQUEST=VERIFYX macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=VERIFYX macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

---

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACROUTE.</td>
</tr>
</tbody>
</table>

RACROUTE
### REQUEST=VERIFYX

One or more blanks must follow RACROUTE.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTINFO=account addr</td>
<td>account addr: A-type address</td>
</tr>
<tr>
<td>APL=applname</td>
<td>applname: 1–8 character name</td>
</tr>
<tr>
<td>APPL=applname addr</td>
<td>applname addr: A-type address</td>
</tr>
<tr>
<td>ENCRYPT=YES</td>
<td>Default: ENCRYPT=YES</td>
</tr>
<tr>
<td>ENCRYPT=NO</td>
<td></td>
</tr>
<tr>
<td>ERROROPT=ABEND</td>
<td>Default: ERROROPT=ABEND</td>
</tr>
<tr>
<td>ERROROPT=NOABEND</td>
<td></td>
</tr>
<tr>
<td>EXENODE=execution node addr</td>
<td>execution node addr: A-type address</td>
</tr>
<tr>
<td>GROUP=group addr</td>
<td>group addr: A-type address</td>
</tr>
<tr>
<td>INSTLN=parm list addr</td>
<td>parm list addr: A-type address</td>
</tr>
<tr>
<td>JOBNAME=jobname addr</td>
<td>jobname addr: A-type address</td>
</tr>
<tr>
<td>LOG=ASIS</td>
<td>Default: LOG=ASIS</td>
</tr>
<tr>
<td>LOG=ALL</td>
<td></td>
</tr>
<tr>
<td>LOGSTR=logstr addr</td>
<td>logstr addr: A-type address</td>
</tr>
<tr>
<td>OIDCARD=oid addr</td>
<td>oid addr: A-type address</td>
</tr>
<tr>
<td>NEWPASS=new password addr</td>
<td>new password addr: A-type address</td>
</tr>
<tr>
<td>NEWPHRASE=new password phrase addr</td>
<td>new password phrase addr: A-type address</td>
</tr>
<tr>
<td>PASSCHK=YES</td>
<td>Default: PASSCHK=YES</td>
</tr>
<tr>
<td>PASSCHK=NO</td>
<td></td>
</tr>
<tr>
<td>PASSWRD=password addr</td>
<td>password addr: A-type address</td>
</tr>
<tr>
<td>PGMNAME=programmer name addr</td>
<td>programmer name addr: A-type address</td>
</tr>
<tr>
<td>PHRASE=password phrase addr</td>
<td>password phrase addr: A-type address</td>
</tr>
<tr>
<td>POE=port of entry addr</td>
<td>port of entry addr: A-type address</td>
</tr>
<tr>
<td>POENET=network name addr</td>
<td>network name addr: A-type address</td>
</tr>
<tr>
<td>REMOTE=YES</td>
<td>Default: REMOTE=NO</td>
</tr>
<tr>
<td>REMOTE=NO</td>
<td></td>
</tr>
<tr>
<td>SECLABL=seclabel addr</td>
<td>seclabel addr: A-type address</td>
</tr>
</tbody>
</table>
### REQUEST=VERIFYX

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>SERVAUTH=servaauth addr</em></td>
<td>servauth addr: A-type address</td>
</tr>
<tr>
<td><em>SESSION=type</em></td>
<td>type: Any valid session type</td>
</tr>
<tr>
<td></td>
<td><strong>Default</strong>: SESSION=TSO</td>
</tr>
<tr>
<td><em>SGROUP=submitting group addr</em></td>
<td>submitting group addr: A-type address</td>
</tr>
<tr>
<td><em>SMC=YES</em></td>
<td>Default: SMC=YES</td>
</tr>
<tr>
<td><em>SMC=NO</em></td>
<td></td>
</tr>
<tr>
<td><em>SNODE=submitting node addr</em></td>
<td>submitting node addr: A-type address</td>
</tr>
<tr>
<td><em>START=procname addr</em></td>
<td>procname addr: A-type address</td>
</tr>
<tr>
<td><em>STAT=ASIS</em></td>
<td>Default: STAT=ASIS</td>
</tr>
<tr>
<td><em>STAT=NO</em></td>
<td></td>
</tr>
<tr>
<td><em>STOKEN=stoken addr</em></td>
<td>stoken addr: A-type address</td>
</tr>
<tr>
<td><em>SUSERID=submitting userid addr</em></td>
<td>submitting userid addr: A-type address</td>
</tr>
<tr>
<td><em>TERMD=terminal addr</em></td>
<td>terminal addr: A-type address</td>
</tr>
<tr>
<td><em>TOKNIN=utoken addr</em></td>
<td>utoken addr: A-type address</td>
</tr>
<tr>
<td><em>TOKNOUT=utoken addr</em></td>
<td>utoken addr: A-type address</td>
</tr>
<tr>
<td><em>TRUSTED=YES</em></td>
<td>Default: TRUSTED=YES</td>
</tr>
<tr>
<td><em>TRUSTED=NO</em></td>
<td></td>
</tr>
<tr>
<td><em>USERID=userid addr</em></td>
<td>userid addr: A-type address</td>
</tr>
<tr>
<td><em>MF=L</em></td>
<td></td>
</tr>
</tbody>
</table>

The parameters are explained under the standard form of the RACROUTE REQUEST=VERIFYX macro instruction, with the following exception:

- _MF=L_
  
  specifies the list form of the RACROUTE REQUEST=VERIFYX macro instruction.

### RACROUTE REQUEST=VERIFYX (execute form)

The execute form of the RACROUTE REQUEST=VERIFYX macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=VERIFYX macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

---

name: Symbol. Begin name in column 1.

b

One or more blanks must precede RACROUTE.

RACROUTE
REQUEST=VERIFYX

b

One or more blanks must follow RACROUTE.

REQUEST=VERIFYX

,RELEASE=number
,RELEASE=(number,CHECK)

,ACTINFO=account addr

,APPL=applname addr

,ENCRIPT=YES
,ENCRIPT=NO

,ERROROPT=ABEND
,ERROROPT=NOABEND

,EXENODE=execution node addr

,GROUP=group addr

,INSTLN=parm list addr

,JOBNAME=jobname addr

,LOG=ASIS
,LOG=ALL

,LOGSTR=logstr addr

,NEWPASS=new password addr

,NEWPHRASE=new password phrase addr

,OIDCARD=oid addr

,PASSCHK=YES
,PASSCHK=NO

,PASSWRD=password addr

,PGMNAME=programmer name addr

,PHRASE=password phrase addr

,POE=port of entry addr

,POENET=network name addr

,REMOTE=YES
The parameters are explained under the standard form of the RACROUTE REQUEST=VERIFYX macro instruction, with the following exceptions:

,MF=(E,ctrl addr)  
specifies the execute form of the RACROUTE REQUEST=VERIFYX macro, using a remote, control-program parameter list.

,RELEASE=number  
,RELEASE=(number,CHECK)  
specifies the RACF release level of the parameter list to be generated by this macro.

To use the parameters associated with a release, you must specify the release number of that release or a later release number. If you specify an earlier release level, the parameter is not accepted by macro processing, and an error message is issued at assembly time.
REQUEST=VERIFYX

When you specify the RELEASE keyword, checking is done at assembly time. Execution-time validation of the compatibility between the list and execute forms of the RACROUTE REQUEST=VERIFYX macro can be done by your specifying the CHECK subparameter on the execute form of the macro.

When CHECK processing is requested, if the size of the list-form expansion is not large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro, the execute form of the macro is not done.

The release specified must be 1.9 or higher.

RACROUTE REQUEST=VERIFYX (modify form)

The modify form of the RACROUTE REQUEST=VERIFYX macro is written as follows. Refer to the standard form of the RACROUTE REQUEST=VERIFYX macro to determine additional parameters that are required by the time the RACROUTE service is invoked using the execute form of the macro.

```
name: Symbol. Begin name in column 1.

b
One or more blanks must precede RACROUTE.

RACROUTE
b
One or more blanks must follow RACROUTE.

REQUEST=VERIFYX

,ACTINFO=account addr
account addr: Rx-type address or register (2) – (12)

,APPL=applname addr
applname addr: Rx-type address or register (2) – (12)

,ENCRYPT=YES
,ENCRYPT=NO

,ERROROPT=ABEND
,ERROROPT=NOABEND
Default: ERROROPT=ABEND

,EXENODE=execution node addr
execution node addr: Rx-type address or register (2) – (12)

,GROUP=group addr
group addr: Rx-type address or register (2) – (12)

,INSTLN=parm list addr
parm list addr: Rx-type address or register (2) – (12)

,JOBNAME=jobname addr
jobname addr: Rx-type address or register (2) – (12)

,LOG=ASIS
,LOG=ALL

,LOGSTR=logstr addr
logstr addr: Rx-type address or register (2) – (12)

,NEWPASS=new password addr
new password addr: Rx-type address or register (2) – (12)
```
### REQUEST=VERIFYX

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEWPHRASE</td>
<td>new password phrase addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>OIDCARD</td>
<td>oid addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>PASSCHK</td>
<td>PASSCHK=YES, PASSCHK=NO</td>
<td></td>
</tr>
<tr>
<td>PASSWRD</td>
<td>password addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>PGMNAME</td>
<td>programmer name addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>PHRASE</td>
<td>password phrase addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>POE</td>
<td>port of entry addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>POENET</td>
<td>network name addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>REMOTE</td>
<td>REMOTE=YES, REMOTE=NO</td>
<td></td>
</tr>
<tr>
<td>SECLABL</td>
<td>seclabel addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>SERVAUTH</td>
<td>servauth addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>SESSION</td>
<td>type: Any valid session type</td>
<td></td>
</tr>
<tr>
<td>SGROUP</td>
<td>submitting group addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>SMC</td>
<td>SMC=YES, SMC=NO</td>
<td></td>
</tr>
<tr>
<td>SNODE</td>
<td>submitting node addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>START</td>
<td>start addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>STAT</td>
<td>STAT=ASIS, STAT=NO</td>
<td></td>
</tr>
<tr>
<td>STOKEN</td>
<td>token addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>SUSERID</td>
<td>submitting userid addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>TLMID</td>
<td>terminal addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>TOKIN</td>
<td>utoken addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>TKNOUT</td>
<td>utoken addr: Rx-type address or register (2) – (12)</td>
<td>Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>TRUSTED</td>
<td>TRUSTED=YES, TRUSTED=NO</td>
<td></td>
</tr>
</tbody>
</table>
REQUEST=VERIFYX

.USERID=userid addr

userid addr: Rx-type address or register (2) – (12)

,MF=(M,ctrl addr)

ctrl addr: Rx-type address or register (1) or (2) – (12)

The parameters are explained under the standard form of the RACROUTE REQUEST=VERIFYX macro instruction, with the following exception:

,MF=(M,ctrl addr)

specifies the modify form of the RACROUTE REQUEST=VERIFYX macro, using a remote, control-program parameter list.
Appendix A. Independent RACF system macros

This appendix contains independent RACF system macros that can be used by other callers to invoke RACF or another external security product.

Attention: As of RACF 1.9, new keywords are not supported on the independent invocation of these macros. RACF supports the new keywords only if you invoke the RACF system macros, which provide more function, using the RACROUTE interface documented in Chapter 2, “RACF system macros,” on page 7.

Table 16 identifies the RACROUTE macro request types that are replacements for the independent system macros described in this appendix. If you receive a return code or reason code from an independent system macro and cannot find a description of the code in this appendix, see the counterpart request type in RACROUTE.

Table 16. Cross-reference for RACROUTE REQUEST=type and the independent RACF system macros

<table>
<thead>
<tr>
<th>RACROUTE REQUEST type</th>
<th>Independent RACF system macro</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQUEST=AUTH</td>
<td>RACHECK</td>
</tr>
<tr>
<td>REQUEST=DEFINE</td>
<td>RACDEF</td>
</tr>
<tr>
<td>REQUEST=EXTRACT</td>
<td>RACXTRT</td>
</tr>
<tr>
<td>REQUEST=FASTAUTH</td>
<td>FRACHECK</td>
</tr>
<tr>
<td>REQUEST=LIST</td>
<td>RAclist</td>
</tr>
<tr>
<td>REQUEST=STAT</td>
<td>RACSTAT</td>
</tr>
<tr>
<td>REQUEST=VERIFY</td>
<td>RACINIT</td>
</tr>
</tbody>
</table>

The following list provides a brief description of the independent RACF system macros.

FRACHECK
Used to provide authorization checking when a user requests access to a RACF-protected resource, similar to RACHECK. However, FRACHECK verifies access to only those resources that have RACF profiles brought into main storage by the RAclist macro service.

RACDEF
Used to define, modify, or delete resource profiles for RACF.

RACHECK
Used to provide authorization checking when a user requests access to a RACF-protected resource.

RACINIT
Used to provide RACF user identification and verification.

RAclist
Used to retrieve general-resource profiles and build an in-storage list for faster authorization checking. The list is attached to the ACEE.

RACSTAT
Used to determine whether RACF is active, and, optionally, to determine
whether RACF protection is in effect for a given resource class. The RACSTAT macro can also be used to determine whether a resource-class name is defined to RACF.

**RACXTRT**

Used to retrieve or update specified resource-profile fields, or to encode data.

The user receives the RACF system macros as part of the MVS product, not as part of RACF. The MVS installation receives these macros even if it does not intend to install RACF.

**FRACHECK macro**

The FRACHECK macro is used to check the authorization for access of a user to a resource. FRACHECK verifies access to those resources whose RACF profiles have been brought into main storage by the RACLIST facility. FRACHECK is a branch-entered service that does not save registers upon entry. Registers 0–5, 14, and 15 are used by the FRACHECK macro instruction and are not restored. Registers 6–13 are not altered by FRACHECK.

**Note:**

1. Do not define profile names containing “RACFVARS”, double asterisks, or internal asterisks in classes that use this macro for authorization. Results are unpredictable.
2. SECLABEL class processing is not done for FRACHECK

**Attention:** The FRACHECK macro executes in the addressing mode of the caller. Therefore, to access profiles that reside above 16MB, the program that issues FRACHECK must be running in 31-bit addressing mode when it issues FRACHECK.

**FRACHECK (standard form)**

The standard form of the FRACHECK macro instruction is written as follows:

```
name name: Symbol. Begin name in column 1.

b One or more blanks must precede FRACHECK.

FRACHECK

b One or more blanks must follow FRACHECK.

ENTITY=entity addr entity addr: A-type address or register (2) – (12)

,CLASS='class name' class name: 1–8 character class name

,CLASS=class name addr class name addr: A-type address or register (2) – (12)

,ATTR=READ

,ATTR=UPDATE

,ATTR=CONTROL

,ATTR=ALTER

,ATTR=reg reg: Registers (2) – (12)
```

270 z/OS V2R1.0 Security Server RACROUTE Macro Reference
The parameters are explained as follows:

**ENTITY=entity addr**

specifies that RACF authorization checking is to be performed for the resource whose name is pointed to by the specified address. The resource name is a 6-byte volume serial number for CLASS='DASDVOL' or CLASS='TAPEVOL'. The name must be left-aligned and padded with blanks. The length of all other resource names is determined from the class descriptor table.

**CLASS='class name'**

specifies that RACF authorization checking is to be performed for a resource of the specified class. If an address is specified, the address must point to an 8-byte field containing the class name.

**ATTR=READ**

**ATTR=UPDATE**

**ATTR=CONTROL**

**ATTR=ALTER**

**ATTR=reg**

specifies the access authority required by the user or group accessing the resource:

- **READ**: RACF user or group can open the resource only to read.
- **UPDATE**: RACF user or group can open the resource to read or write.
- **CONTROL**: For MVS/VSAM data sets, RACF user or group has authority equivalent to the VSAM control password. For non-VSAM data sets and other resources, RACF user or group has UPDATE authority.
- **ALTER**: RACF user or group has total control over the resource.

If a register is specified, the register must contain one of the following codes in the low-order byte of the register:

- **X'02'** READ
- **X'04'** UPDATE
- **X'08'** CONTROL
- **X'80'** ALTER

**ACEE=acee addr**

specifies the address of the ACEE to be used to check authorization and to locate the in-storage profiles (RACLST output) for the specified classes. If an ACEE is specified, it is used for authorization checking. If the specified ACEE has an in-storage profile list for the specified class, it is used to locate the resource. If an ACEE is not specified or there is no in-storage profile list for the specified class in the ACEE, RACF uses the TASK ACEE (TCBSENV) pointer in the extended TCB. Otherwise, or if the TASK ACEE pointer is zero,
FRACHECK

RACF uses the main ACEE for the address space to obtain the list of the in-storage profiles. The main ACEE is pointed to by the ASXBSENV field of the address-space extension block.

\texttt{WKAREA=area addr}

specifies the address of a 16-word work area to be used by FRACHECK. It contains the following information:

- Word 12 contains the reason code that RACF passes back to the FRACHECK caller in register 0.
- Word 13 contains the return code that FRACHECK passes back to the caller in register 15.
- Word 14 contains the address of the in-storage profile used to determine authorization, or zero if no profile is found. If the profile was found, the profile address is passed back to the caller in register 1.
- Word 15 contains a value provided by a preprocessing installation exit, or zero if there is no preprocessing exit.

\texttt{APPL='applname'}

\texttt{APPL=applname addr}

specifies the name of the application requesting the authorization checking. This information is not used for the authorization-checking process, but is made available to the installation exit or exits. If an address is specified, it should point to an 8-byte area containing the application name, left-justified and padded with blanks if necessary.

\texttt{INSTLN=parm list addr}

specifies the address of an area that contains information for the FRACHECK installation exit. This address is passed to the exit routine when the exit is given control. The INSTLN parameter is used by application or installation programs to pass information to the FRACHECK installation exit.

Parameters for RELEASE=1.6 through 1.8.1

The following parameters are valid for RELEASE=1.6, RELEASE=1.7, RELEASE=1.8, and RELEASE=1.8.1. RELEASE=1.6 is the default.

- ACEE=
- APPL=
- ATTR=
- CLASS=
- ENTITY=
- INSTLN=
- RELEASE=
- WKAREA=

Return codes and reason codes

For FRACHECK, if the return codes and reason codes you are receiving are not discussed in the description of this macro, refer to the return codes and reason codes described with RACROUTE REQUEST=FASTAUTH on page 144.

When control is returned, register 15 contains one of the following return codes, and register 0 can contain a reason code:

<table>
<thead>
<tr>
<th>Hex code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The user or group is authorized to use the resource.

**Reason code**

**Meaning**

0  The FRACHECK return code indicates whether the caller is authorized to access the resource, but the access attempt is not within the scope of the audit or global audit specification.

4  The FRACHECK return code indicates whether the caller is authorized to access the resource, but the access attempt is within the scope of the audit or global audit specification. The FRACHECK caller should log the attempt by issuing a RACHECK for the resource that the caller is attempting to access.

The resource or class name is not defined to RACF.

08  The user or group is not authorized to use the resource.

**Reason code**

**Meaning**

0  The FRACHECK return code indicates whether the caller is authorized to access the resource, but the access attempt is not within the scope of the audit or global audit specification.

4  The FRACHECK return code indicates whether the caller is authorized to access the resource, but the access attempt is within the scope of the audit or global audit specification. The FRACHECK caller should log the attempt by issuing a RACHECK for the resource that the caller is attempting to access.

0C  RACF is not active.

10  FRACHECK installation-exit error occurred.

14  RACF is not installed or an insufficient level of RACF is installed.

18  Indicates the profile has a conditional access list, the port-of-entry field in the security token is blank-filled, and the port-of-entry class is active.

64  Indicates that the CHECK subparameter of the RELEASE keyword was specified on the execute form of the FRACHECK macro; however, the list form of the macro does not have the proper RELEASE parameter. Macro processing terminates.

### FRACHECK (list form)

The list form of the FRACHECK macro instruction is written as follows:

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>name: Symbol. Begin name in column 1.</td>
</tr>
<tr>
<td>b</td>
<td>One or more blanks must precede FRACHECK.</td>
</tr>
<tr>
<td>FRACHECK</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>One or more blanks must follow FRACHECK.</td>
</tr>
<tr>
<td>ENTITY=entity addr</td>
<td>entity addr: A-type address.</td>
</tr>
<tr>
<td><code>CLASS=</code>class name`</td>
<td>class name: 1–8 character class name</td>
</tr>
</tbody>
</table>
### FRACHECK (execute form)

The execute form of the FRACHECK macro instruction is written as follows:

```
name: Symbol. Begin name in column 1.
b
FRACHECK

b

ENTITY=entity addr

CLASS=class name addr

ATTR=reg

ACEE=acee addr

WKAREA=area addr

APPL=applname addr

INSTLN=parm list addr

RELEASE=number

RELEASE=(number, CHECK)
```

- **name**: Symbol. Begin name in column 1.
- **b**: One or more blanks must precede FRACHECK.
- **FRACHECK**: One or more blanks must follow FRACHECK.

### Parameters

- **ENTITY=entity addr**: Rx-type address or register (2) – (12)
- **CLASS=class name addr**: Rx-type address or register (2) – (12)
- **ATTR=reg**: Rx-type address or register (2) – (12)
- **ACEE=acee addr**: Rx-type address or register (2) – (12)
- **WKAREA=area addr**: Rx-type address or register (2) – (12)
- **APPL=applname addr**: Rx-type address or register (2) – (12)
- **INSTLN=parm list addr**: Rx-type address or register (2) – (12)
- **RELEASE=number**: 1.8.1, 1.8, 1.7, or 1.6
- **RELEASE=(number, CHECK)**
  - **Default**: RELEASE=1.6
The parameters are explained under the standard form of the FRACHECK macro instruction with the following exceptions:

\[MF=(E, \text{ctrl addr})\]

specifies the execute form of the FRACHECK macro instruction, using a remote, control-program parameter list.

\[RELEASE=\text{number}\]
\[RELEASE=(, \text{CHECK})\]
\[RELEASE=(\text{number}, \text{CHECK})\]

specifies the RACF release level of the parameter list to be generated by the macro.

To use the parameters associated with a release, you must specify the number of that release or a later release number. If you specify an earlier release level, the parameter is not accepted by macro processing, and an error message is issued at assembly time. For the parameters that are valid for RELEASE=1.6 and later, see “Parameters for RELEASE=1.6 through 1.8.1” on page 272.

When you specify the RELEASE keyword, checking is done at assembly time. Compatibility between the list and execute forms of the FRACHECK macro are validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.

**RACDEF: Define a resource to RACF**

The RACDEF macro is used to define, modify, or delete resource profiles for RACF. It can also be used for special cases of authorization checking. RACF uses the resulting profiles to perform RACHECK authorization checking. The RACDEF caller must be authorized (APF-authorized, in system key 0–7, or in supervisor state).

A RACF user can change or add the RACDEF parameters, OWNER, LEVEL, UACC, or AUDIT by means of the RACDEF preprocessing and postprocessing exit routines.

**Note:** Only callers in 24-bit addressing mode can issue this macro. Callers executing in 31-bit addressing mode, who want to use the RACDEF function, can code the RACROUTE macro.

When activated, automatic direction of application updates propagates RACDEF requests that update the RACF database on to selected remote nodes.

Not all RACDEF requests update the RACF database. The ENVIR=VERIFY keyword specifies that no profile is to be created, but that the user’s authority is to be checked. Automatic direction of application updates does not propagate a RACDEF if ENVIR=VERIFY is specified. Similarly, many RACDEF requests are issued with RACFIND=NO to check if a user is authorized to create a data set or catalog a data set based on a generic profile. RACDEF RACFIND=NO requests for
DASD data sets are not propagated. RACDEF requests for tape data sets are propagated even when RACFIND=NO is specified because even though no data set profile is updated, an update might be made to the TVTOC in a TAPEVOL profile.

Only RACDEF requests with return code 0 are propagated.

## RACDEF (standard form)

The standard form of the RACDEF macro is written as follows:

```
name name

name: Symbol. Begin name in column 1.

b

One or more blanks must precede RACDEF.

RACDEF

b

One or more blanks must follow RACDEF.

ENTITY=profile name addr

profile name addr: A-type address, or register (2) – (12)

,VOLSER=vol addr

vol addr: A-type address, or register (2) – (12)

Note: VOLSER is required only for CLASS='DATASET' and DSTYPE not equal to M when a discrete profile name is used.

,TYPE=DEFINE

,new dsn addr: A-type address, or register (2) – (12)

,TYPE=ADDVOL,

,old vol addr: A-type address, or register (2) – (12)

,TYPE=CHGVOL,

,old vol addr: A-type address, or register (2) – (12)

,TYPE=DELETE

Default: TYPE=DEFINE

,DDNAME='ddname'

ddbname: 1-8 character name

,DDNAME=ddname addr

ddbname addr: A-type address or register (2) – (12)

Note: DDNAME is valid for OPEN/CLOSE routine.

,DSTYPE=V

Default: DSTYPE=N

,DSTYPE=V

,DSTYPE=M

,DSTYPE=T

,INSTLN=parm list addr

parm list addr: A-type address, or register (2) – (12)

,CLASS='class name'

class name: 1-8 character class name

,CLASS=class name addr

class name addr: A-type address, or register (2) – (12)

Default: CLASS='DATASET'

,MENTITY=entity addr

entity addr: A-type address, or register (2) – (12)

,MCLASS='class name'

class name: 1-8 character class name

,MCLASS=class name

class name addr: A-type address, or register (2) – (12)
ADDR

\texttt{MVOLSER=volser \textit{addr}} \quad \textit{volser \textit{addr}}: A-type address, or register (2) – (12)

\texttt{MGENER=ASIS}
\texttt{MGENER=YES}

\texttt{ACCE=acce \textit{addr}} \quad \textit{acce \textit{addr}}: A-type address, or register (2) – (12)

\texttt{UNIT=unit \textit{addr}} \quad \textit{unit \textit{addr}}: A-type address, or register (2) – (12)

\texttt{OWNER=owner \textit{id} \textit{addr}} \quad \textit{owner \textit{id} \textit{addr}}: A-type address, or register (2) – (12)

\texttt{LEVEL=number}
\texttt{LEVEL=\textit{reg}} \quad \textit{reg}: Register (2) – (12)

\texttt{UACC=ALTER}
\texttt{UACC=CONTROL}
\texttt{UACC=UPDATE}
\texttt{UACC=READ}

\texttt{UACC=EXECUTE}
\texttt{UACC=NONE}
\texttt{UACC=\textit{reg}} \quad \textit{reg}: Register (2) – (12)

\texttt{DATA=data \textit{addr}} \quad \textit{data \textit{addr}}: A-type address or register (2) – (12)

\texttt{AUDIT=NONE}
\texttt{AUDIT=\textit{audit value}}
\texttt{AUDIT=\textit{audit value} (access level)}
\texttt{AUDIT=\textit{audit value} (access level),...} \quad \textit{access level}: READ, UPDATE, CONTROL, or ALTER
\texttt{AUDIT=} \quad \texttt{\textit{reg}} \quad \textit{reg}: Register (2) – (12)

\texttt{RACFIND=YES}
\texttt{RACFIND=NO}

\texttt{CHKAUTH=YES}
\texttt{CHKAUTH=NO}

\texttt{GENERIC=YES}
\texttt{GENERIC=ASIS}

\texttt{WARNING=YES}
\texttt{WARNING=NO}

\texttt{FILESEQ=\textit{reg}} \quad \textit{reg}: Register (2) – (12)
\texttt{FILESEQ=\textit{number}} \quad \textit{number}: 1–65535

\texttt{EXPDT=expir-date \textit{addr}} \quad \textit{expir-date \textit{addr}}: A-type address or register (2) – (12)

\texttt{EXPDTX=extended expir-date \textit{addr}} \quad \textit{extended expir-date \textit{addr}}: A-type address or register (2) – (12)

\texttt{RETPD=retu-period \textit{addr}} \quad \textit{retu-period \textit{addr}}: A-type address or register (2) – (12)
\texttt{RETPD=} \quad \texttt{Default: See description of parameter.}
The parameters are explained as follows:

**ENTITY=profile name addr**
specifies the address of the name of the discrete or generic profile that is to be defined to, modified, or deleted from RACF. The profile name is a 44-byte DASD data set name for CLASS='DATASET' or a 6-byte volume-serial name for CLASS='DASDVOL' or CLASS='TAPEVOL'. The lengths of all other profile names are determined by the class descriptor table. The name must be left-justified in the field and padded with blanks.

**VOLSER=vol addr**
specifies the address of the volume-serial number:
- For TYPE=ADDVOL, of the new volume to be added to the definition of the data set.
- For TYPE=ADDVOL and CLASS='TAPEVOL', of the new volume being added to the tape-volume set identified by ENTITY.
- For TYPE=DEFINE and CLASS='DATASET', of the catalog (for a VSAM data set), or of the volume on which the data set resides (for a non-VSAM data set).

The volume serial number is optional if DSTYPE=M is specified; it is ignored if the profile name is generic.

The field pointed to by the specified address contains the volume serial number (padded to the right with blanks, if necessary, to make six characters).

```
ACCLVL=(access value addr)  access value addr: A-type address or register (2) – (12)
ACCLVL=(access value addr)  access value addr: A-type address or register (2) – (12)
ACCLVL=(access value addr parm list addr)  parm list addr: A-type address, or register (2) – (12)
TAPELBL=STD  Default: TAPELBL=STD
TAPELBL=BLP
TAPELBL=NL
SECLVL=addr  addr: A-type address, or register (2) – (12)
ERASE=YES  Default: ERASE=NO
NOTIFY=notify-id addr  notify-id addr: A-type address or register (2) – (12)
ENVIR=VERIFY  Specifies that only verification is to be done.
             Default: Normal RACDEF processing.
RESOWN=resource owner addr  A-type address, or register (2) – (12)
STORCLA=storage class addr  A-type address, or register (2) – (12)
MGMTCLA=management type addr  A-type address, or register (2) – (12)
```
RACDEF

!,TYPE=ADDVOL, OLDVOL= old vol addr
!,TYPE=CHGVOL, OLDVOL= old vol addr
!,TYPE=DELETE

specifies the type of action to be taken:

- TYPE=DEFINE. The definition of the resource is added to the RACF data set, and the current user is established as the owner of the defined entity.
- TYPE=DEFINE,NEWNAME=. If NEWNAME is specified, the address points to a 44-byte field containing the new name for the data set that is to be renamed. NEWNAME is valid only with CLASS=DATASET. NEWNAME is not valid with DSTYPE=T.
- TYPE=ADDVOL. The new volume is added to the definition of the specified resource. For the DATASET class, the OLDVOL address specifies a previous volume of a multivolume data set. For the TAPEVOL class, the ENTITY address specifies a previous volume of a tape volume set. This parameter applies only to discrete profiles.
- TYPE=CHGVOL. The volume serial number in the definition of the specified resource is changed from the old volume serial number identified in OLDVOL to the new volume serial number identified in the VOLSER parameter. This parameter applies only to discrete profiles. TYPE=CHGVOL is not valid with DSTYPE=T.
- TYPE=DELETE. The definition of the resource is removed from the RACF data set. (For a multivolume data set or a tape volume set, only the specified volume is removed from the definition.)

If DSTYPE=T is specified, the data sets must be deleted in reverse of the order in which they were created. For example, if file1 has data-set1, file2 has data-set2, and file3 has data-set3, you must do the RACDEF TYPE=DELETE, DSTYPE=T for file3, file2, and file1, in that order.

Note:

- If SETROPTS ADDCREATOR is in effect when a new DATASET or general resource profile is defined, the profile creator’s user ID is placed on the profile access list with ALTER authority.
- If SETROPTS NOADDCREATOR is in effect when a new generic profile is defined, the profile creator’s user ID is not placed on the profile’s access list. If you use profile modeling when defining a generic profile, RACF copies the access list exactly. If the creator’s user ID appeared in the model’s access list, the same authority is copied to the new profile.
- If SETROPTS NOADDCREATOR is in effect when a new discrete DATASET or TAPEVOL profile is defined, the profile creator’s user ID is placed on the profile’s access list with ALTER authority. If you use profile modeling when defining one of these profiles, if the creator’s user ID appeared in the model’s access list, the authority is created in the new profile with ALTER authority.
- If SETROPTS NOADDCREATOR is in effect when any other new discrete profile is defined, the profile’s creator’s user ID is not placed on the access list. If you use profile modeling when defining one of these profiles, RACF copies the access list exactly. If the creator’s user ID appeared in the model’s access list, the same authority is copied to the new profile.

!,DDNAME= 'ddname'
!,DDNAME= ddname addr

specifies the ddname associated with the data set name that is specified as the ENTITY or ENTITYX name.

This keyword is intended for use only by OPEN processing.
DSTYPE specifies the type of data set associated with the request:

- N for non-VSAM
- V for VSAM
- M for model profile
- T for tape.

DSTYPE=T should not be specified unless the SETROPTS TAPEDSN option is active (RCVTTDSN bit is on); otherwise, the processing is the same as for RACDEF CLASS='TAPEVOL'.

Specify DSTYPE only for CLASS='DATASET'.

INSTLN parm list addr

Specifies the address of an area that is to contain parameter information meaningful to the RACDEF installation exit routines. This information is passed to the installation exit routines when they are given control from the RACDEF routine.

The INSTLN parameter can be used by an application program acting as a resource manager that needs to pass information to the RACDEF installation exit routines.

CLASS='class name'

Specifies that a profile is to be defined, modified, or deleted in the specified class. If an address is specified, the address must point to a 1-byte length field followed by the class name (for example, DATASET or TAPEVOL). The class name should be no longer than eight characters.

MCLASS='class name'

Specifies the class to which the profile defined by MENTITY= belongs. If an address is specified, the address must point to a 1-byte length field followed by the class name. The class name should be no longer than eight characters. The default is MCLASS='DATASET'.

MVOLSER=volser addr

Specifies the address of the volume serial number of the volume associated with the profile in the ENTITY operand. The field pointed to by the specified address contains the volume serial number, padded to the right with blanks if necessary to make six characters.

If you specify MENTITY and CLASS='DATASET', you must specify MVOLSER with the name of the volume serial number or with blanks.
If you specify it with blanks, the discrete MENTITY data set profile name must be unique, meaning it has no duplicates on the database. In this case, RACF determines the correct MVOLSER.

`MGENER=ASIS`
`MGENER=YES`

specifies whether the profile name defined by MENTITY is to be treated as a generic name.

- If MGENER=ASIS is specified, the profile name is considered a generic only if it contains a generic character: an asterisk (*) or a percent sign (%).
- If MGENER=YES is specified, the profile name is considered a generic, even if it does not contain a generic character: an asterisk (*) or a percent sign (%). If you specify MGENER=YES, the resource name in the macro will match only a generic resource name in the RACF data base. It will not match a discrete name.

MGENER is ignored if the GENCMD option on the RACF SETROPTS command is not specified for the class. (See z/OS Security Server RACF Command Language Reference.)

`ACEE=acee addr`

specifies the address of the ACEE to be used during RACDEF processing.

If no ACEE is specified, RACF uses the TASK ACEE pointer (TCBSENV) in the extended TCB. If the TASK ACEE pointer is zero, RACF uses the main ACEE. The main ACEE's address is in the ASXBSENV field in the address space extension block.

`UNIT=unit addr`

specifies the address of a field containing unit information. If a unit address is specified, the unit information in the data set profile is replaced by the unit information pointed to by this unit address. The unit address must point to a field containing a 1-byte length field (whose value can range from 4 through 8) followed by the actual unit information. If the value in the length field is 4, the unit information is assumed to contain a copy of the information in the UCBTYP field of the UCB. Otherwise the unit information is assumed to be in the generic form (for example, 3330-1).

UNIT is valid if TYPE=CHGVOL or TYPE=DEFINE is specified. It is ignored for generic names.

`OWNER=owner id addr`

specifies the address of a field containing the profile owner's ID. OWNER is valid if TYPE=DEFINE is specified. The owner's ID must be a valid (RACF-defined) user ID or group name. The address must point to an 8-byte field containing the owner's name, left-justified and padded with blanks.

`LEVEL=number`
`LEVEL=reg`

specifies a level value for the profile. LEVEL is valid only if TYPE=DEFINE is specified. The level number must be a valid decimal number in the range 0 to 99. If a register is specified, its low-order byte must contain the binary representation of the number.

Note: RACF does not check the validity of this number if it has been added or modified by the RACDEF preprocessing or postprocessing exit routines.

`UACC=ALTER`
`UACC=CONTROL`
`UACC=UPDATE`
.UACC=READ
.UACC=EXECUTE
.UACC=NONE
.UACC=reg

specifies a universal-access authority for the profile. UACC is valid only if TYPE=DEFINE is specified. UACC must contain a valid access authority (EXECUTE, ALTER, CONTROL, UPDATE, READ, or NONE).

For EXECUTE to be effective, your MVS system must be running with DFP 3.1.0 or later installed. EXECUTE authority means that the user has only the ability to execute the program; the user cannot read the program. If your MVS system is not running with DFP 3.1.0 or later an EXECUTE UACC specification is treated as NONE.

If a register is specified, the low-order byte must contain one of the following valid access authorities:

- X'80' ALTER
- X'40' CONTROL
- X'20' UPDATE
- X'10' READ
- X'01' NONE
- X'08' EXECUTE

Note: RACF does not check the validity of the universal-access authority if it has been added or modified by the RACDEF preprocessing or postprocessing exit routine.

.DATA=data addr

specifies the address of a field that contains up to 255 characters of installation-defined data to be placed in the profile. DATA is valid only if TYPE=DEFINE is specified. The data address must point to a field containing a 1-byte length field (whose value can range from 0 to 255) followed by the actual installation-defined data.

.AUDIT=NONE
.AUDIT=audit value
.AUDIT=(audit value(access level),audit value(access level),...)
.AUDIT=(reg)

specifies the types of accesses and the access levels that are to be logged to the SMF data set. AUDIT is valid only if TYPE=DEFINE is specified.

For audit value, specify one of the following: ALL, SUCCESS, or FAILURES. You can, optionally, specify an access level (access authority) following each audit value.

Access Levels:
- EXECUTE: This access level is not audited. If you specify FAILURES (READ), logs the READ attempt as a failure, but allows EXECUTE access to the data set.
- READ: The default access level value, logs access attempts at any level.
- UPDATE: Logs access attempts at the UPDATE, CONTROL, and ALTER levels.
- CONTROL: Logs access attempts at the CONTROL and ALTER levels.
- ALTER: Logs access attempts at the ALTER level only.
**Note:** For more information about specific audit values and access levels, see z/OS Security Server RACF Command Language Reference.

RACF resolves combinations of conflicting specifications by using the most encompassing specification. Thus, in the case of the following example:

```
ALL(UPDATE), FAILURES(READ)
```

RACF assumes `SUCCESS(UPDATE), FAILURES(READ)`.

For compatibility with previous releases, register notation can also be specified as `AUDIT=reg` if the register is not given the symbolic name ALL, SUCCESS, or FAILURES.

Logging is controlled separately for SUCCESS and FAILURES, and can also be suppressed or requested using the RACHECK postprocessing installation exit routine.

If a register is specified, its low-order byte must contain one of the following valid audit values:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ALL</td>
</tr>
<tr>
<td>1</td>
<td>SUCCESS</td>
</tr>
<tr>
<td>2</td>
<td>FAILURES</td>
</tr>
<tr>
<td>3</td>
<td>NONE</td>
</tr>
<tr>
<td>4-5</td>
<td>Qualifier for SUCCESS</td>
</tr>
<tr>
<td>6-7</td>
<td>Qualifier for FAILURES</td>
</tr>
</tbody>
</table>

The qualifier codes are as follows:

```
00  READ
01  UPDATE
10  CONTROL
11  ALTER
```

Only one of bits 0 through 3 can be on. If ALL is specified, the two qualifier fields can be used to request different logging levels for successful and unsuccessful events.

```
,RACFIND=YES
,RACFIND=NO
```

specifies whether a discrete profile is involved in RACDEF processing. When TYPE=DEFINE is specified, RACFIND=YES means that a discrete profile is to be created. When TYPE=DELETE, DEFINE with NEWNAME, CHGVOL, or ADDVOL is specified, RACFIND=YES means that a discrete profile already exists.

RACFIND=NO means (when TYPE=DEFINE) that no discrete profile is to be created, but some authorization checking is required. For other types of action, no discrete profile should exist.

**Note:**

1. Automatic direction of application updates does not propagate a RACDEF request for DASD data sets when the RACFIND=NO keyword is specified.
2. Automatic direction of application updates does not propagate a RACDEF request for tape data sets when the RACFIND=NO keyword is specified. Even though no DATASET class profile is updated, the TAPEVOL class profile is updated.

,CHKAUTH=YES
,CHKAUTH=NO

specifies whether or not RACF verifies that the user is authorized to perform the operation.

CHKAUTH=YES is valid when either TYPE=DEFINE,NEWNAME= or TYPE=DELETE is specified.

For DSTYPE=T, specifies that RACF verifies that the user is authorized to define a data set (TYPE=DEFINE), delete a data set (TYPE=DELETE), or add a volume (TYPE=ADDVOL).

,FILESEQ=number
,FILESEQ=reg

specifies the file sequence number of a tape data set on a tape-volume or within a tape-volume set. The number must be in the range 1–65535. If a register is specified, it must contain the file-sequence number in the low-order halfword. If CLASS='DATASET' and DSTYPE=T are not specified, FILESEQ is ignored.

,EXPDT=expir-date addr
,RETPD=retn-period addr
,EXPDTX=extended expir-date addr

specifies the address containing information about the data set's expiration date or RACF security retention period.

EXPDT=expir-date addr specifies the address of a 3-byte field containing the data set's expiration date. The date is given in packed decimal form as YYDDDF, where YY is the year and DDD is the day number. The year must be in the range 01 through 99, and the day number must be in the range 1 through 366. F allows the date to remain a positive integer when converted from packed decimal to hexadecimal. All fields are right-justified.

EXPDTX=extended expir-date addr specifies the address of a 4-byte field that contains the address of the data set's expiration date. The date is given in packed decimal form as CCYYDDDF, where CC is the century change greater than 19, YY is the year, and DDD is the day number. The year must be in the range 01 through 99. The day must be in the range 1 through 366. All fields are right-justified. When you want to represent 19 for the century, you must specify CC as 00; when you want to represent 20 for the century, you must specify CC as 01. F allows the date to remain a positive integer when converted from packed decimal to hexadecimal. To use this parameter, you must also specify RELEASE=1.8.

RETPD=retn-period addr specifies the address of a two-byte binary field containing the number of days after which RACF protection for the data set expires. The value specified must be in the range 1 through 65533. To indicate that there is no expiration date, specify 65534.

If you do not specify any of these parameters, a default RACF security retention period is obtained from the RETPD keyword specified on an earlier RACF SETROPTS command.

These parameters are valid only if CLASS='DATASET' and DSTYPE=T.
,ACCLVL=(access value addr, parm list addr)
specifies the tape-label access-level information for the MVS tape-label
functions. The address must point to a field containing a 1-byte length field
(with a value that can range from 0 through 8) followed by an 8-character
string that is passed to the RACDEF installation exit routines. The
parameter-list address points to a parameter list containing additional
information to be passed to the RACDEF installation exit routines.
RACF does not check or modify this information.
TAPELBL=STD|BLP|NL
specifies the type of tape label processing to be done:
  STD   IBM or ANSI standard labels
  BLP   Bypass label processing
  NL    Unlabeled tapes

For TAPELBL=BLP, the user must have the requested authority to the profile
ICHBLP in the general-resource class FACILITY. For TAPELBL=NL or BLP, the
user is not allowed to protect volumes with volume serial numbers in the
format Lnnnnn.

The TAPELBL parameter is passed to the RACDEF installation exits.
This parameter is primarily intended for use by data-management routines to
indicate the label type from the LABEL keyword on the JCL statement.
This parameter is valid only for CLASS='DATASET' and DSTYPE=T, or
CLASS='TAPEVOL'.

,SECLVL=addr
specifies the address of a list of installation-defined security-level identifiers.
Each identifier is a halfword, containing a value that corresponds to an
installation-defined security-level name.
The identifiers must be in the range 1 through 254. Only one identifier can be
passed in the list.
The list must start with a fullword containing the number of entries in the list
(currently, only 0 or 1).

,ERASE=YES
,ERASE=NO
specifies whether the DASD data set, or the released space, is to be erased
when it is deleted or part of its space is to be released for reuse.
• If ERASE=YES is specified, the data set is erased when it is deleted, or
released for reuse.
• If ERASE=NO is specified, the data set is not be erased, deleted, or released.

Note: This parameter can be overridden by the RACF SETROPTS ERASE
command.
The default is ERASE=NO.
Specify ERASE only for CLASS=DATASET.
Only programs that are APF-authorized, system key 0–7, or in supervisor state
can use the ERASE keyword.

,NOTIFY=notify-id addr
specifies the address of an 8-byte area containing the user ID of the
RACDEF

RACF-defined user who is to be notified when an unauthorized attempt to access the resource protected by this profile is detected.

**.GENERIC=YES .GENERIC=ASIS**

specifies whether the resource name is treated as a generic profile name. If GENERIC is specified with CLASS=DEFINE, NEWNAME, GENERIC applies to both the old and new names. GENERIC is ignored if the GENCMD option on the RACF SETROPTS command is not specified for the class. (See z/OS Security Server RACF Command Language Reference.)

This keyword is designed primarily for use by RACF commands.

- If GENERIC=YES is specified, the resource name is considered a generic profile name, even if it does not contain a generic character: an asterisk (*) or a percent sign (%). If you specify GENERIC=YES, the resource name in the macro will match only a generic resource name in the RACF database. It will not match a discrete name.

- If GENERIC=ASIS is specified, the resource name is considered a generic only if it contains a generic character: an asterisk (*) or a percent sign (%).

**.WARNING=YES .WARNING=NO**

WARNING is valid only if TYPE=DEFINE is specified. If WARNING=YES is specified, access is granted to the resource and the event is logged as a warning if either the SUCCESS or FAILURES logging is requested.

**.ENVIR=VERIFY**

specifies that no profile is to be created, but that the user's authority to define or rename the resource or profile is to be checked, along with any other authorization processing that is necessary.

If you specify ENVIR, you must also specify RELEASE=1.8.1 or a later release number.

**Note:**

1. If you do not specify ENVIR=VERIFY, normal RACDEF processing occurs.
2. Automatic direction of application updates does not propagate RACDEF request if ENVIR=VERIFY is specified.

**.RESOWN=resource owner address**

specifies the address of a field containing the resource owner's ID. If you specify RESOWN, you must also specify TYPE=DEFINE and the current RELEASE parameter. The resource owner's ID must be either a valid (RACF-defined) user ID or group name, or *NONE*. If the resource owner's ID is specified as *NONE*, RACF performs third-party RACHECK, using USERID=*NONE*. The address must point to a 2-byte field followed by the resource owner's name.

**.MGMTCLA=management class address**

specifies the address of a management class to which the resource owner must have authority. The address must point to an 8-byte field that contains a management-class name preceded by a halfword length. If you specify MGMTCLA, you must also specify TYPE=DEFINE, RESOWN, and RELEASE=1.8.1 or a later release number.

When MGMTCLA is specified, RACDEF processing invokes RACHECK processing to verify that the RESOWNER is authorized to the management class.
STORCLA=storage class address

specifies the address of the storage class to which the resource owner must have authority. The address must point to a 2-byte field followed by the storage class name. If you specify STORCLA, you must also specify TYPE=DEFINE, RESOWN, and RELEASE=1.8.1 or a later release number.

When specified, RACDEF processing invokes RACHECK processing to verify that the RESOWNER is authorized to the storage class.

Parameters for RELEASE=1.6 through 1.8.1

The RELEASE values for which a parameter is valid are marked with an ‘X’. RELEASE=1.6 is the default.

Table 17. RACDEF parameters for RELEASE=1.6 through 1.8.1

<table>
<thead>
<tr>
<th>Parameter</th>
<th>RELEASE=1.6</th>
<th>RELEASE=1.7</th>
<th>RELEASE=1.8 or 1.8.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACEE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ACCLVL=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>AUDIT=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CHKAUTH=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CLASS=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DATA=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DSTYPE=N, V, or M</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DSTYPE=T</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENTITY=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENVIR=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ERASE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>EXPDT=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>EXPDXT=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>FILESEQ=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>GENERIC=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>INSTLN=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LEVEL=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>MCLASS=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>MENTITY=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>MGENER=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>MGMTCLA=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>MVOLSER=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NOTIFY=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RACFIND=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RELEASE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RESOWN=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RETPD=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SECLVL=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>STORCLA=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TAPELBL=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
Table 17. RACDEF parameters for RELEASE=1.6 through 1.8.1 (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>RELEASE=1.6</th>
<th>RELEASE=1.7</th>
<th>RELEASE=1.8 or 1.8.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>UACC=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>UNIT=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VOLSER=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>WARNING=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Return codes and reason codes

If the return codes and reason codes you are receiving are not discussed in this macro, refer to the return codes and reason codes described with RACROUTE REQUEST=DEFINE "Return codes and reason codes" on page 73.

When control is returned, register 15 contains one of the following return codes, and register 0 can contain a reason code.

Hex code

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
</tr>
</tbody>
</table>

Reason code

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
</tr>
<tr>
<td>04</td>
</tr>
</tbody>
</table>

04

RACDEF has completed processing. Register 0 contains one of the following reason codes:

Reason code

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
</tr>
<tr>
<td>-</td>
</tr>
<tr>
<td>-</td>
</tr>
<tr>
<td>-</td>
</tr>
<tr>
<td>04</td>
</tr>
</tbody>
</table>

08

RACDEF has completed processing. Register 0 contains one of the following reason codes:

Reason code

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
</tr>
</tbody>
</table>
- For TYPE=DEFINE, the check for authority to allocate a data set or create a profile with the specified name has been failed.

- For TYPE=DELETE or TYPE=DEFINE,NEWNAME if CHKAUTH=YES is specified, the authorization check has been failed.

- For TYPE=ADDVOL,OLDVOL (or for TYPE=CHGVOL,OLDVOL) the old value was not defined.

04 Indicates for TYPE=DEFINE that no profile was found to protect the data set and that the RACF protect-all option is in effect.

08 Indicates TYPE=DEFINE (or TYPE=ADDVOL,OLDVOL or TYPE=CHGVOL,OLDVOL) and DSTYPE=T were specified, and the user is not authorized to define a data set on the specified volume.

0C Indicates TYPE=DEFINE and DSTYPE=T were specified, and the user is not authorized to define a data set with the specified name.

10 Indicates DSTYPE=T or CLASS=TAPEVOL was specified, and the user is not authorized to specify LABEL=(,(BLP)).

20 Indicates the data set owner is not authorized to use the specified DFP storage class.

24 Indicates the data set owner is not authorized to use the specified DFP management class.

0C For TYPE=DEFINE,NEWNAME, the old data set name was not defined; or if the generation-data-group (GDG) modeling function is active, an attempt was made to rename a GDG name to a name that requires the creation of a new profile; or if generic profile checking is active, the old data set name was protected by a generic profile and there is no generic profile that protects the new data set name. This last case refers only to an attempt to rename an existing profile, which cannot be found.

10 For TYPE=DEFINE with MENTITY, the model resource was not defined.

64 Indicates that the CHECK subparameter of the RELEASE keyword was specified on the execute form of the RACDEF macro; however, the list form of the macro does not have the proper RELEASE parameter. Macro processing terminates.

Example 1
Invoke RACF to define a discrete profile for a non-VSAM data set residing on the volume pointed to by register 8. Register 7 points to the data set name. All successful requests for update authority to the data set are to be audited, as well as all unsuccessful ones.

```
RACDEF ENTITY=(R7),VOLSER=(R8),CLASS='DATASET',
       AUDIT=(SUCCESS(UPDATE),FAILURES),
       RACFIND=YES
```

Example 2
Use the standard form of the RACDEF macro to define a discrete data set profile for a non-VSAM DASD data set. The data set for which you are creating a profile is a non-VSAM DASD data set named DSNAME. It resides on a volume ID named VOLID. You want to create a discrete profile by specifying the RACFIND keyword. In addition, you want to notify the user called USERNAME of any access attempts that have been rejected because they exceed the UACC of READ that you are allowing.
RACDEF

RACDEF ENTITY=DSNAME,VOLSER=VOLID,CLASS='DATASET',UACC=READ, RACFIND=YES,NOTIFY=USERNAME,RELEASE=1.7

Example 3
Use the standard form of the macro to check the authority of a user to define a discrete data set profile for a non-VSAM DASD data set, but do not actually build the profile. The name of the data set is DSNAME.
RACDEF ENTITY=DSNAME,VOLSER=VOLID,CLASS='DATASET',RACFIND=NO

Example 4
Use the standard form of the macro to define a generic data set profile protecting a resource named PROFNAME. Use the discrete profile protecting a resource named MDELPROF whose volser is in MDELVOL as a model for the new profile. Notify the user named USERNAME of any access attempts that have been rejected because they exceed the UACC of READ that you are allowing.
RACDEF ENTITY=PROFNAME,CLASS='DATASET',GENERIC=YES,ENTITY=MDELPROF, MVOLSER=MDELVOL, UACC=READ, NOTIFY=USERNAME, RELEASE=1.7

Example 5
Use the standard form of the macro to define a tape-volume profile for a volume whose ID is VOLID. Allow a universal-access level of READ.
RACDEF ENTITY=VOLID,CLASS='TAPEVOL', UACC=READ

Example 6
Use the standard form of the macro to delete a discrete data set profile named DSNAME, located on the volume named VOLID.
RACDEF TYPE=DELETE,ENTITY=DSNAME,VOLSER=VOLID,CLASS='DATASET'

RACDEF (list form)
The list form of the RACDEF macro is written as follows:

name: Symbol. Begin name in column 1.

/bslash
RACDEF
/bslash

One or more blanks must precede RACDEF.

ENTITY=profile name addr: A-type address.

Note: ENTITY must be specified on either the list or the execute form of the macro.

,VOLSER=vol addr: A-type address

Note: VOLSER is required (on either LIST or EXECUTE) only for CLASS='DATASET' and DSTYPE not equal to M when a discrete profile name is used.

,TYPE=DEFINE
,TYPE=DEFINE,
NEWNAME=
new dsn addr: A-type address

,TYPE=ADDVOL,
OLDVOL=
old vol addr: A-type address

,TYPE=CHGVOL,
OLDVOL=
old vol addr

(TYPE=DELETE)

,DDNAME='ddname'
,DDNAME=ddname

Note: DDNAME is valid for OPEN/CLOSE routine.

,DSTYPE=N
,DSTYPE=V
,DSTYPE=M
,DSTYPE=T

,INSTLN=parm list addr

,CLASS='class name'
,CLASS=class name

Default: CLASS='DATASET'

,MENTITY=entity addr

,MCLASS='class name'
,MCLASS=class name

Default: MCLASS='DATASET'

,MVOLSER=volser addr

,MGENER=ASIS
,MGENER=YES

,ACEE=acee addr

,UNIT=unit addr

,OWNER=owner id addr

,LEVEL=number

Default: Zero.

,UACC=ALTER
,UACC=CONTROL
,UACC=UPDATE
,UACC=READ
,UACC=EXECUTE
,UACC=NONE

,DATA=data addr

,AUDIT=NONE
,AUDIT=audit value
,AUDIT=(audit value)

Default: READ

,RACFIND=YES
,RACFIND=NO

,CHKAUTH=YES
,CHKAUTH=NO

Default: CHKAUTH=NO
### RACDEF

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GENERIC=YES</td>
<td>Default: GENERIC=ASIS</td>
</tr>
<tr>
<td>GENERIC=ASIS</td>
<td></td>
</tr>
<tr>
<td>WARNING=YES</td>
<td>Default: WARNING=NO</td>
</tr>
<tr>
<td>WARNING=NO</td>
<td>Note: Warning is valid only if TYPE=DEFINE is specified.</td>
</tr>
<tr>
<td>FILESEQ=number</td>
<td>number: 1–65535</td>
</tr>
<tr>
<td>EXPDT=expir-date addr</td>
<td>expir-date addr: A-type address</td>
</tr>
<tr>
<td>RETPD=retm-period addr</td>
<td>retm-period addr: A-type address</td>
</tr>
<tr>
<td>EXPDTX=extended expir-date addr</td>
<td>extended expir-date addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>ENVIR=VERIFY</td>
<td>Specifies that only verification is to be done. Default: Normal RACDEF processing.</td>
</tr>
<tr>
<td>RESOWN=resource owner addr</td>
<td>resource owner addr: A-type address</td>
</tr>
<tr>
<td>STORCLA=storage class addr</td>
<td>storage class addr: A-type address</td>
</tr>
<tr>
<td>MGMTCLA=management type</td>
<td>management type: A-type address Default: See description of parameter.</td>
</tr>
<tr>
<td>ACCLVL=(access value addr)</td>
<td>access value addr: A-type address</td>
</tr>
<tr>
<td>ACCLVL=(access value addr, parm list addr)</td>
<td>parm list addr: A-type address</td>
</tr>
<tr>
<td>TAPELBL=STD</td>
<td>Default: TAPELBL=STD</td>
</tr>
<tr>
<td>TAPELBL=BLP</td>
<td></td>
</tr>
<tr>
<td>TAPELBL=NL</td>
<td></td>
</tr>
<tr>
<td>SECLVL=addr</td>
<td>addr: A-type address</td>
</tr>
<tr>
<td>ERASE=YES</td>
<td>Default: ERASE=NO</td>
</tr>
<tr>
<td>ERASE=NO</td>
<td></td>
</tr>
<tr>
<td>NOTIFY=notify-id addr</td>
<td>notify-id addr: A-type address</td>
</tr>
<tr>
<td>MF=L</td>
<td></td>
</tr>
</tbody>
</table>

The parameters are explained under the standard form of the RACDEF macro instruction with the following exception:

**`MF=L`**

specifies the list form of the RACDEF macro instruction.

### RACDEF (execute form)

The execute form of the RACDEF macro is written as follows:

```plaintext
name

b
```

name: Symbol. Begin name in column 1.

One or more blanks must precede RACDEF.
RACDEF

RACDEF

♭ One or more blanks must follow RACDEF.

___________________________________________________________________________________

ENTITY=profile name addr profile name addr: Rx-type address

Note: ENTITY must be specified on either the list or the execute form of the macro.

,,VOLSER=vol addr vol addr: Rx-type address or register (2) – (12)

Note: VOLSER is required only for CLASS=’DATASET’ and DSTYPE not equal to M when a discrete profile name is used.

,,TYPE=DEFINE

new dsn addr: Rx-type address or register (2) – (12)

,,TYPE=DEFINE,

NEWNAME=

new dsn addr

,,TYPE=ADDIVOL,

OLDVOL=

old vol addr

,,TYPE=CHGVOL,

OLDVOL=

old vol addr

,,TYPE=DELETE

,,DDNAME=ddname addr ddname addr: A-type address or register (2) – (12)

Note: DDNAME is valid for OPEN/CLOSE routine.

,,DSTYPE=N

,,DSTYPE=V

,,DSTYPE=M

,,DSTYPE=T

,,INSTLN=parm list addr parm list addr: Rx-type address or register (2) – (12)

,,CLASS=class name addr class name addr: Rx-type address or register (2) – (12)

,,ENTITY=entity addr entity addr: Rx-type address or register (2) – (12)

,,MENTITY=entity addr entity addr: Rx-type address or register (2) – (12)

,,MCCLASS=class name addr class name addr: Rx-type address or register (2) – (12)

,,MVOLSER=volser addr volser addr: Rx-type address or register (2) – (12)

,,MGENER=ASIS

,,MGENER=YES

,,ACCE=acce addr acce addr: Rx-type address or register (2) – (12)

,,UNIT=unit addr unit addr: Rx-type address or register (2) – (12)

,,OWNER=owner id addr owner id addr: Rx-type address or register (2) – (12)

,,LEVEL=number

,,LEVEL=reg reg: Register (2) – (12)

,,UACC=ALTER

,,UACC=CONTROL

,,UACC=UPDATE

,,UACC=READ

Appendix A. Independent RACF system macros 293
RACDEF

,UACC=EXECUTE
,UACC=None
,UACC=reg

.DATA=data addr

,AUDIT=None
,AUDIT=audit value
,AUDIT=(audit value (access level), audit value (access level))
,AUDIT=(reg)

,RACFIND=YES
,RACFIND=NO

,CHKAUTH=YES
,CHKAUTH=NO

,GENERIC=YES
,GENERIC=NO

,WARNING=YES
,WARNING=NO

,RELEASE=number
,RELEASE=(CHECK)
,RELEASE=(number, CHECK)

,FILESEQ=reg
,FILESEQ=number

,EXPDT=expir-date addr
,RETPD=retn-period addr
,EXPDTX=extended expir-date addr

,ACCLVL=(access value addr,parm list addr)

,ENVIR=VERIFY

,RESOWN=resource owner addr

,STORCLA=storage class addr

,MGMTCLA=management class addr

,TAPELBL=STD
,TAPELBL=BLP
,TAPELBL=NL

Note: Warning is valid only if TYPE=DEFINE is specified.

Default: RELEASE=1.6

number: 1.6, 1.7, or 1.8

reg: Register (2) – (12)

data addr: Rx-type address or register (2) – (12)

audit value: ALL, SUCCESS, or FAILURES

access level: READ, UPDATE, CONTROL, or ALTER

reg: Register (2) – (12)

number: 1–65535

expir-date addr: Rx-type address or register (2) – (12)

retn-period addr: Rx-type address or register (2) – (12)

extended expir-date addr: A-type address or register (2) – (12)

resource owner addr: Rx-type address or register (2) – (12)

storage class addr: Rx-type address or register (2) – (12)

management class addr: Rx-type address or register (2) – (12)

access value addr: Rx-type address or register (2) – (12)

parm list addr: Rx-type address or register (2) – (12)

Specifies that only verification is to be done.

Rx-type address or register (2) – (12)

Rx-type address or register (2) – (12)

Rx-type address or register (2) – (12)

Rx-type address or register (2) – (12)

Rx-type address or register (2) – (12)

Rx-type address or register (2) – (12)
The parameters are explained under the standard form of the RACDEF macro instruction with the following exceptions:

\textbf{.MF=(E,ctrl addr)}

specifies the execute form of the RACDEF macro, using a remote, control-program parameter list.

\textbf{.RELEASE=number}

\textbf{.RELEASE=(,CHECK)}

\textbf{.RELEASE=(number, CHECK)}

specifies the RACF release level of the parameter list to be generated by this macro.

To use the parameters associated with a release, you must specify the release number of that release or a later release number. If you specify an earlier release level, the parameter is not accepted by macro processing, and an error message is issued at assembly time. For the parameters that are valid for RELEASE=1.6 and later, see Table 17 on page 287.

The default is RELEASE=1.6.

When you specify the RELEASE keyword, checking is done at assembly time. Compatibility between the list and execute forms of the RACDEF macro is validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.

**RACHECK: Check RACF authorization**

The RACHECK macro is used to provide authorization checking when a user requests access to a RACF-protected resource. These parameters are restricted in use and must be used only by programs that are authorized (APF-authorized, in system key 0–7, or in supervisor state).

\textbf{Note:} Only callers in 24-bit addressing mode can issue this macro. Callers executing in 31-bit addressing mode who want to use the RACHECK function can code the RACROUTE macro.

**RACHECK (standard form)**

The standard form of the RACHECK macro is written as follows:

\begin{verbatim}
name
\end{verbatim}
### RACHECK

One or more blanks must precede RACHECK.

**RACHECK**

One or more blanks must follow RACHECK.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROFILE=profile addr</td>
<td>profile addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>ENTITY=(resource name addr)</td>
<td>resource name addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>ENTITY=(resource name addr,CFA)</td>
<td>resource name addr,CFA: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>ENTITY=(resource name addr,PRIVATE)</td>
<td>resource name addr,PRIVATE: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>ENTITY=(resource name addr,NONE)</td>
<td>resource name addr,NONE: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>VOLSER=vol addr</td>
<td>vol addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>Note: VOLSER is required only for CLASS='DATASET' and DSTYPE not equal to M when a discrete profile name is used and only when ENTITY is also coded.</td>
<td></td>
</tr>
<tr>
<td>CLASS='class name'</td>
<td>class name: 1–8 character class name</td>
</tr>
<tr>
<td>CLASS=class name addr</td>
<td>class name addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>ATTR=READ</td>
<td>reg: Register (2) – (12)</td>
</tr>
<tr>
<td>Default: ATTR=READ</td>
<td></td>
</tr>
<tr>
<td>ATTR=UPDATE</td>
<td></td>
</tr>
<tr>
<td>ATTR=CONTROL</td>
<td></td>
</tr>
<tr>
<td>ATTR=ALTER</td>
<td></td>
</tr>
<tr>
<td>ATTR=reg</td>
<td></td>
</tr>
<tr>
<td>DDNAME='ddname'</td>
<td>ddname: 1–8 character name</td>
</tr>
<tr>
<td>DDNAME=ddname addr</td>
<td>ddname addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>Note: DDNAME is valid for OPEN/CLOSE routine only.</td>
<td></td>
</tr>
<tr>
<td>DSTYPE=N</td>
<td>Default: DSTYPE=N</td>
</tr>
<tr>
<td>DSTYPE=V</td>
<td></td>
</tr>
<tr>
<td>DSTYPE=M</td>
<td></td>
</tr>
<tr>
<td>DSTYPE=T</td>
<td></td>
</tr>
<tr>
<td>INSTLN=parm list addr</td>
<td>parm list addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>LOG=ASIS</td>
<td>Default: LOG=ASIS</td>
</tr>
<tr>
<td>LOG=NOFAIL</td>
<td></td>
</tr>
<tr>
<td>LOG=NONE</td>
<td></td>
</tr>
<tr>
<td>LOG=STAT</td>
<td></td>
</tr>
<tr>
<td>OLDVOL=old vol addr</td>
<td>old vol addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>APPL='applname'</td>
<td>applname: 1–8 character name</td>
</tr>
<tr>
<td>APPL=applname addr</td>
<td>applname addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>ACEE=acce addr</td>
<td>acce addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>ACCLVL=(access value addr)</td>
<td>access value addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>ACCLVL=(access value addr,parm list addr)</td>
<td>access value addr,parm list addr: A-type address or register (2) – (12)</td>
</tr>
<tr>
<td>RACFIND=YES</td>
<td></td>
</tr>
<tr>
<td>RACFIND=NO</td>
<td></td>
</tr>
</tbody>
</table>
The parameters are explained as follows:

- **PROFILE=profile addr**
- **ENTITY=(resource name addr)**
- **ENTITY=(resource name addr, CSA)**
- **ENTITY=(resource name addr, PRIVATE)**
- **ENTITY=(resource name addr, NONE)**

- **PROFILE=profile addr** specifies that RACF authorization checking is to be performed for the resource whose profile is pointed to by the specified address. This profile must be supplied by ENTITY=(xxx, CSA). A profile supplied by RACLIST is not acceptable. To specify PROFILE, programs must be APF-authorized and in supervisor state. The programs must also be in system key 0 or in the same key as the storage of the profile.

For the ENTITY keyword, the resource name is a 44-byte DASD data set name for CLASS='DATASET', or a 6-byte volume serial number for CLASS='DASDVOL' or CLASS='TAPEVOL'. The length of all other resource names is determined from the class descriptor table.

- ENTITY=(resource name addr) specifies that RACF authorization checking is to be performed for the resource whose name is pointed to by the specified address. The name must be left-justified in the field and padded with blanks.

- ENTITY=(resource name addr, CSA) specifies that RACF authorization checking is to be performed for the indicated resource, and that a copy of the profile is to be maintained in main storage. The storage acquired for the profile is obtained from the common storage area (CSA), and is fetch-protected, key 0 storage. The issuer of RACHECK must free this storage when the profile is no longer needed. (The profile subpool number and length are part of the profile data returned.) If CSA is specified and the return code produced by the RACHECK macro instruction is 00 or 08, the address of the profile is returned in register 1.

Only programs that are APF-authorized, system key 0–7, or in supervisor state can use the CSA parameter.
By establishing and maintaining a resource profile, the resource manager can reduce the I/O required to perform RACF authorization checks on frequently accessed resources.

- ENTITY=(resource name addr,PRIVATE) PRIVATE specifies the same as CSA except that RACHECK returns the profile in the user's private area rather than in common storage, and the name field contains the name of the returned profile instead of the name of the resource that was specified on the ENTITY keyword. The issuer of RACHECK must free this storage when the profile is no longer needed. (The profile subpool number and length are returned as well as the profile data.) Only programs that are APF-authorized, system key 0–7, or in supervisor state can use the PRIVATE parameter.

- ENTITY=(resource name addr,NONE) specifies the same as ENTITY=resource name address. However, no profile is returned.

, VOLSER=vol addr
   specifies the volume serial number, as follows:
   - For MVS/VSAM DASD data sets, this is the volume serial number of the catalog controlling the data set.
   - For non-VSAM DASD data sets and tape data sets, this is the volume serial number of the volume on which the data set resides.

   The volume serial number is optional if DSTYPE=M is specified; it is ignored if the profile name is generic.

   The field pointed to by the specified address contains the volume serial number, padded to the right with blanks if necessary to make six characters. VOLSER= is only valid and must be supplied with CLASS='DATASET', (unless DSTYPE=M is specified) and if ENTITY is also coded.

, CLASS='class name'

, CLASS=class name addr
   specifies that RACF authorization checking is to be performed for a resource of the specified class. If an address is specified, the address must point to a 1-byte field indicating the length of the class name, followed by the class name.

, ATTR=READ
, ATTR=UPDATE
, ATTR=CONTROL
, ATTR=ALTER
, ATTR=reg
   specifies the access authority of the user or group permitted access to the resource for which RACF authorization checking is to be performed:
   - READ: RACF user or group can open the resource only to read.
   - UPDATE: RACF user or group can open the resource to write or read.
   - CONTROL: For VSAM data sets, RACF user or group has authority equivalent to the VSAM control password. For non-VSAM data sets and other resources, RACF user or group has UPDATE authority.
   - ALTER: RACF user or group has total control over the resource.

   If a register is specified, the register must contain one of the following codes in the low-order byte of the register:

   X'02' READ
   X'04' UPDATE
   X'08' CONTROL
RACHECK

X'80'  ALTER

,DDNAME='ddname'
,DDNAME='ddname addr'

specifies the ddname associated with the data set name that is specified as the
ENTITY name.

This keyword is intended for use only by OPEN processing.

,DSTYPE=N
,DSTYPE=V
,DSTYPE=M
,DSTYPE=T

specifies the type of data set associated with the request:

- N for non-VSAM
- V for VSAM
- M for model profile
- T for tape.

DSTYPE=T should not be specified unless the SETROPTS TAPEDSN option is
active (RVTTDSN bit is on); otherwise, the processing is the same as for
RACHECK CLASS='TAPEVOL'.

DSTYPE should be specified only for CLASS='DATASET'.

,INSTLN=parm list addr

specifies the address of an area that is to contain parameter information
meaningful to the RACHECK installation exit routine. This information is
passed to the installation exit routine when it is given control by RACHECK.

The INSTLN parameter can be used by an application program acting as a
resource manager that needs to pass information to the RACHECK installation
exit routine.

,LOG=ASIS
,LOG=NOFAIL
,LOG=NONE
,LOG=NOSTAT

specifies the types of access attempts to be recorded on the SMF data set.

- ASIS: RACF records the event in the manner specified in the profile that
  protects the resource.
- NOFAIL: If the authorization check fails, the attempt is not recorded. If the
  authorization check succeeds, the attempt is recorded as in ASIS.
- NONE: The attempt is not to be recorded.
  LOG=NONE suppresses both messages and SMF records regardless of
  MSGSUPP=NO.
- NOSTAT: The attempt is not to be recorded. No logging is to occur, and no
  resource statistics are to be updated (including messages and SMF records).

Only programs that are APF-authorized, system key 0–7, or in supervisor state,
can use the LOG parameter.

,OLDVOL=old vol addr

specifies a volume serial:

- For CLASS='DATASET', within the same multivolume data set specified by
  VOLSER=.
- For CLASS='TAPEVOL', within the same tape volume specified by
  ENTITY=.
RACF authorization checking verifies that the OLDVOL specified is part of the same multivolume data set or tape-volume set.

The specified address points to the field that contains the volume serial number, padded to the right with blanks if necessary to make six characters.

\texttt{,APPL='applname'}

\texttt{,APPL=applname addr}

specifies the name of the application requesting authorization checking. The \texttt{applname} is not used for the authorization-checking process but is made available to the installation exit routine or routines called by the RACHECK routine. If the address is specified, the address must point to an 8-byte field containing the application name, left-justified and padded with blanks.

\texttt{,ACEE=acee addr}

specifies the address of the ACEE to be used during RACHECK processing. If no ACEE is specified, RACF uses the TASK ACEE pointer (TCBSENV) in the extended TCB. If the TASK ACEE pointer is zero, RACF uses the main ACEE for the address space. The main ACEE is pointed to by the ASXBSENV field of the address-space extension block.

Only programs that are APF-authorized, system key 0–7, or in supervisor state, can use the ACEE parameter.

\texttt{,ACCLVL=(access value addr)}

\texttt{,ACCLVL=(access value addr,parm list addr)}

specifies the tape-label access-level information for the MVS tape-label functions. The access value pointed to by the specified address is a 1-byte length field, containing the value (0–8) of the length of the following data, followed by an 8-character string that is passed to the RACHECK installation exit routines. The optional parameter list pointed to by the specified address contains additional information to be passed to the RACHECK installation exit routines. RACF does not inspect or modify this information.

\texttt{,RACFIND=YES}

\texttt{,RACFIND=NO}

indicates whether or not the resource is protected by a discrete profile. The RACF processing and the possible return codes are given in Table 18 on page 301.

\textbf{Note:} In all cases, a return code of X'0C' is also possible if the OLDVOL specified was not part of the multivolume data set defined by Volser, or it was not part of the same tape volume defined by ENTITY.

\texttt{,GENERIC=YES}

\texttt{,GENERIC=ASIS}

specifies whether the resource name is to be treated as a generic profile name. If GENERIC is specified with CLASS=DEFINE, NEWNAME, then GENERIC applies to both the old and new names. GENERIC is ignored if the GENCMD option on the RACF SETROPTS command is not specified for the class. (See \texttt{z/OS Security Server RACF Command Language Reference}.)

This keyword is designed primarily for use by RACF commands.

- If GENERIC=\texttt{YES} is specified, the resource name is considered a generic profile name, even if it does not contain a generic character: an asterisk (\texttt{*}) or a percent sign (\texttt{%}). If you specify GENERIC=\texttt{YES}, the resource name in the macro will match only a generic resource name in the RACF database. It will not match a discrete name.
If GENERIC=ASIS is specified, the resource name is considered a generic only if it contains a generic character: an asterisk (*) or a percent sign (%).

Table 18. Types of profile checking performed by RACHECK

<table>
<thead>
<tr>
<th>Operand</th>
<th>Generic profile checking inactive</th>
<th>Generic profile checking active</th>
</tr>
</thead>
<tbody>
<tr>
<td>RACFIND=YES</td>
<td>Look for discrete profile; if found, exit with return code 00 or 08. If no discrete profile is found, exit with return code 08.</td>
<td>Look for discrete profile; if found, exit with return code 00 or 08. Look for generic profile; if found, exit with return code 00 or 08. Exit with return code 08 if neither a discrete nor a generic profile is found.</td>
</tr>
<tr>
<td>RACFIND=NO</td>
<td>No checking. Exit with return code 04. (See note).</td>
<td>Look for generic profile; if found, exit with return code 00 or 08. If not found, exit with return code 04. (See note).</td>
</tr>
<tr>
<td>RACFIND not specified</td>
<td>Look for discrete profile; if found, exit with return code 00 or 08. If no discrete profile is found, exit with return code 04. (See note).</td>
<td>Look for discrete profile; if found, exit with return code 00 or 08. Look for generic profile; if found, exit with return code 00 or 08. Exit with return code 04 if neither a discrete nor a generic profile is found. (See note).</td>
</tr>
</tbody>
</table>

Note: If PROTECTALL is active, no profile is found, and the user ID whose authority was checked does not have the SPECIAL attribute, RACF returns a return code X’08’ instead of a return code X’04’ and denies access.

,,FILESEQ=number
,,FILESEQ=reg

specifies the file-sequence number of a tape data set on a tape volume or within a tape-volume set. The value must be in the range 1–65535. If a register is specified, it must contain the file sequence number in the low-order halfword. If CLASS=’DATASET’ and DSTYPE=T are not specified, FILESEQ is ignored.

,,TAPELBL=STD|BLP|NL

specifies the type of tape-label processing to be done:

STD IBM or ANSI standard labels
BLP Bypass label processing
NL Unlabeled tapes

For TAPELBL=BLP, the user must have the requested authority to the profile ICHBLP in the general-resource class FACILITY. For TAPELBL=NL or BLP, the user is not allowed to protect volumes with volume serial numbers in the format “Lnnnnn”.

This parameter is primarily intended for use by data-management routines to indicate the label type from the LABEL keyword on the JCL statement.

This parameter is valid only for CLASS=’DATASET’ and DSTYPE=T, or CLASS=’TAPEVOL’.
RACHECK

,STATUS=NONE|ERASE
specifies whether or not RACHECK is to return the erase status of the given data set. This parameter is valid only for CLASS=’DATASET’ and a DSTYPE value other than T.

,USERID=’userid’
,USERID=user ID addr
specifies the user ID that RACF uses to perform third-party RACHECK. This is an 8-character field that is left-justified and padded to the right with blanks.

If USERID is specified when the caller invokes RACHECK, RACF verifies that user's authority to the given entity; RACF disregards the user ID associated with the ACEE of the caller.

For third-party RACHECK, RACF performs the following steps:
1. Checks to see whether the USERID keyword is *NONE* and GROUPID is not specified. If so, then RACF creates a default user (null) ACEE, which it uses to perform the RACHECK.
2. If not, checks to see whether an additional (third-party) ACEE already exists, chained off the current caller's ACEE or the ACEE specified in the ACEE= keyword.
3. If so, checks to see whether the user ID in that ACEE matches the one specified on the USERID keyword. If so, RACHECK uses the existing ACEE and avoids RACINIT processing.
4. If USERID is specified and RACHECK does not find an additional (third-party) ACEE, or the user ID in the ACEE does not match the user ID specified on the USERID keyword, then RACHECK creates a third-party ACEE based on the USERID keyword.
5. If the GROUPID keyword is specified in addition to the USERID keyword, and a third-party ACEE already exists, the group name of the existing third-party ACEE must also match the group name specified on the GROUPID keyword. If the GROUPID keywords do not match, RACHECK creates a third-party ACEE based on the USERID keyword.

Note: If the calling program does not specify the GROUPID keyword, the internal RACINIT function uses the default group associated with the specified user ID.

Only programs that are APF-authorized, system key 0–7, or in supervisor state, can use the USERID and GROUPID keywords.

Note: If the user ID is *NONE* and a GROUPID has not been specified, then a default user (null) ACEE is created and used to satisfy RACHECK processing.

,GROUPID=’groupid’
,GROUPID=groupid addr
specifies the group name that RACF uses to perform third-party RACHECK.

If the calling program wants a third-party RACHECK performed on the group name rather than the user ID, the USERID keyword must be specified as *NONE*. When the caller invokes third-party RACHECK, RACF verifies the authority of the group name to the requested resource; RACF disregards the group name associated with the ACEE of the caller. For third-party RACHECK, RACF performs the following steps:
- Checks to see whether an additional (third-party) ACEE already exists, chained off the caller's ACEE, or the ACEE specified in the ACEE= keyword.
• If so, checks to see whether the group name matches that specified on the GROUPID keyword. If so, RACHECK uses that ACEE and avoids RACINIT processing.
• If GROUPID is specified and RACHECK does not find an additional (third-party) ACEE, or the group name in the ACEE does not match the group ID specified on the GROUPID keyword, RACHECK creates a third-party ACEE based on the GROUPID keyword.

Only programs that are APF-authorized, system key 0–7, or in supervisor state, can use the USERID and GROUPID keywords.

Parameters for RELEASE=1.6 through 1.8.2

The RELEASE values for which a specific parameter is valid are marked with an ‘X’. RELEASE=1.6 is the default.

Table 19. RACHECK parameters for RELEASE=1.6 through 1.8.2

<table>
<thead>
<tr>
<th>Parameter</th>
<th>RELEASE=1.6</th>
<th>RELEASE=1.7</th>
<th>RELEASE=1.8, 1.8.1, or 1.8.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACEE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ACCLVL=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>APPL=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ATTR=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CLASS=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DSTYPE=N, V, or M</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DSTYPE=T</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>ENTITY=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>FILESEQ=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>GENERIC=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>GROUPID=</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>INSTLN=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LOG=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>OLDVOL=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>OWNER=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PROFILE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RACFIND=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RELEASE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>STATUS=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TAPELBL=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>USERID=</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>VOLSER=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Return codes and reason codes

If the return codes and reason codes you are receiving are not discussed in this macro, refer to the return codes and reason codes described with RACROUTE REQUEST=AUTH on page 44.

When control is returned, register 15 contains one of the following return codes, and register 0 can contain a reason code.
RACHECK

Hex Code

Meaning

00  The user is authorized by RACF to obtain use of a RACF-protected
     resource. Register 0 contains one of the following reason codes:

Reason Code

Meaning

00  Indicates a normal completion.

04  Indicates STATUS=ERASE was specified and the data set is to be
     erased when scratched.

     Otherwise, indicates that the warning status of the resource was
     requested by the RACHECK issuer setting bit X'10' at offset 12
     decimal in the RACHECK parameter list and the resource is in
     warning mode.

10  When CLASS=TAPEVOL, indicates the TAPEVOL profile contains
     a TVTOC.

20  When CLASS=TAPEVOL, indicates that the TAPEVOL profile can
     contain a TVTOC, but currently does not (for a scratch pool
     volume).

24  When CLASS=TAPEVOL, indicates that the TAPEVOL profile does
     not contain a TVTOC.

04  The specified resource is not protected by RACF. Register 0 contains the
     following reason code:

Reason code

Meaning

00  Indicates a normal completion.

08  The user is not authorized by RACF to obtain use of the specified
     RACF-protected resource. Register 0 contains the following reason code:

     00  Indicates a normal completion. A possible cause would be
         PROTECTALL is active, no profile is found, and the user ID whose
         authority was checked does not have the SPECIAL attribute.

     04  Indicates STATUS=ERASE was specified and the data set is to be
         erased when scratched.

     08  Indicates DSTYPE=T or CLASS='TAPEVOL' was specified and the
         user is not authorized to use the specified volume.

     0C  Indicates the user is not authorized to use the data set.

     10  Indicates DSTYPE=T or CLASS='TAPEVOL' was specified and the
         user is not authorized to specify LABEL=(,BLP).

     1C  User with EXECUTE authority to the data set profile specified
         ATTR=READ, and RACF failed the access attempt.

     0C  The OLDVOL specified was not part of the multivolume data set defined
         by VOLSER, or it was not part of the same tape volume defined by
         ENTITY.

     10  RACINIT issued by third-party RACHECK failed. Register 0 contains the
         RACINIT return code.

     64  Indicates that the CHECK subparameter of the RELEASE keyword was
specified on the execute form of the RACHECK macro; however, the list form of the macro does not have the proper RELEASE parameter. Macro processing terminates.

Example 1
Perform RACF authorization checking using the standard form, for a non-VSAM data set residing on the volume pointed to by register 8. Register 7 points to the data set name, and the RACF user is requesting the highest level of control over the data set. The “RACF-indicated” bit in the data set’s DSCB is on. Logging and statistics updates are not to be done.

RACHECK ENTITY=((R7)),VOLSER=(R8),CLASS='DATASET', ATTR=ALTER,RACFIND=YES,LOG=NOSTAT

Example 2
Perform RACF authorization checking using the standard form, for a non-VSAM data set controlled by the catalog pointed to by register 8. Register 7 points to the data set name, and the RACF user is requesting the highest level of control over the data set. The “RACF-indicated” bit in the data set’s DSCB is on.

RACHECK ENTITY=((R7)),VOLSER=(R8),CLASS='DATASET', ATTR=ALTER,RACFIND=YES

Example 3
Perform RACF authorization checking using the standard form, for a VSAM data set residing on the volume pointed to by register 8. Register 7 points to the data set name, and the RACF user is requesting the data set for Read only. Register 4 points to an area containing additional parameter information.

RACHECK ENTITY=((R7)),VOLSER=(R8),CLASS='DATASET', DSTYPE=V,INSTLN=(R4)

Example 4
Using the standard form, perform RACF authorization checking for a tape volume for Read access only. The tape volume is pointed to by register 8 and the volume’s access level is in register 5.

RACHECK ENTITY=((R8)),CLASS='TAPEVOL',ATTR=READ, ACCLVL=((R5))

Example 5
Using the standard form, perform third party RACF authorization checking for a data set for Read access only for a user. Register 7 points to the data set name, and register 8 points to the volume on which the data set resides.

RACHECK ENTITY=((R7)),VOLSER=(R8),CLASS='DATASET', ATTR=READ,USERID='SOMEUSER'

Example 6
Using the standard form, perform third-party RACF authorization checking for a data set for Read access only for a group. Register 7 points to the data set name, and register 8 points to the volume on which the data set resides.

RACHECK ENTITY=((R7)),VOLSER=(R8),CLASS='DATASET',ATTR=READ USERID='*NONE*',GROUPID='ANYGROUP'

Example 7
Using the standard form, perform third-party RACF authorization checking for a data set for a user connected to a group. Register 7 points to the data set name, and register 8 points to the volume on which the data set resides.

RACHECK ENTITY=((R7)),VOLSER=(R8),CLASS='DATASET',ATTR=READ USERID='SOMEUSER',GROUPID='ANYGROUP'
Example 8
Using the standard form, perform third-party RACF authorization checking for a
data set (with Read-only access). Register 7 points to the data set name, and
register 8 points to the volume on which the data set resides. A is an 8-byte
declared field padded with zeros.
RACHECK ENTITY=((R7)), VOLSER=(R8),CLASS='DATASET',
ATTR=READ,USERID=A

RACHECK (list form)
The list form of the RACHECK macro is written as follows:

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>b</td>
</tr>
<tr>
<td></td>
<td>One or more blanks must precede RACHECK.</td>
</tr>
<tr>
<td>RACHECK</td>
<td></td>
</tr>
<tr>
<td></td>
<td>b</td>
</tr>
<tr>
<td></td>
<td>One or more blanks must follow RACHECK.</td>
</tr>
</tbody>
</table>

PROFILE=profile addr
ENTITY=(resource name addr)
ENTITY=(resource name addr,CSA)
ENTITY=(resource name addr,PRIVATE)
ENTITY=(resource name addr,NONE)

VOLSER=vol addr
Note: VOLSER is required on either the list or the execute form of the macro, but only for
CLASS='DATASET' and DSTYPE not equal to M when a discrete profile name is used. If required,
VOLSER must be specified on either the list or the execute form of the macro.

CLASS='class name'
CLASS=class name addr

ATTR=READ
ATTR=UPDATE
ATTR=CONTROL
ATTR=ALTER

DDNAME='ddname'
DDNAME=ddname addr
Note: DDNAME is valid for OPEN/CLOSE routine.

DSTYPE=N
DSTYPE=V
DSTYPE=M
DSTYPE=T

INSTLN=parm list addr

LOG=ASIS
LOG=NOFAIL
LOG=NONE

Default: LOG=ASIS
The parameters are explained under the standard form of the RACHECK macro with the following exception:

\texttt{,MF=L}

specifies the list form of the RACHECK macro instruction.

**RACHECK (execute form)**

The execute form of the RACHECK macro is written as follows:

```
name name: Symbol. Begin name in column 1.

b One or more blanks must precede RACHECK.

RACHECK

b One or more blanks must follow RACHECK.
```
### RACHECK

**PROFILE=** `profile addr`  
*profile addr*: Rx-type address or register (2) – (12)

**ENTITY=(** `resource name addr` **)**  
*resource name addr*: Rx-type address or register (2) – (12)

**ENTITY=(** `resource name addr,CSA` **)**  
*Note: PROFILE or ENTITY is required on either the list or the execute form of the macro.*

**ENTITY=(** `resource name addr,PRIVATE` **)**

**ENTITY=(** `resource name addr,NONE` **)**

**VOLSER=** `vol addr`  
*vol addr*: Rx-type address or register (2) – (12)

*Note: VOLSER is required on either the list or the execute form of the macro, but only for CLASS=’DATASET’ and DSTYPE not equal to M when a discrete profile name is used. If required, VOLSER must be specified on either the list or the execute form of the macro.*

**CLASS=** `class name addr`  
*class name addr*: Rx-type address or register (2) – (12)

**RELEASE=** `number`  
*number*: 1.8.2, 1.8.1, 1.8, 1.7, or 1.6

**RELEASE=(,** `CHECK` **)**  
*Default: RELEASE=1.6*

**RELEASE=** `(number,**CHECK**)`

**ATTR=** `READ`  
*reg*: Register (2) – (12)

**ATTR=** `UPDATE`  

**ATTR=** `CONTROL`  

**ATTR=** `ALTER`  

**ATTR=** `reg`

**DDNAME=** `ddname addr`  
*ddname addr*: Rx-type address or register (2) – (12)

*Note: DDNAME is valid for OPEN/CLOSE routine.*

**DSTYPE=N**

**DSTYPE=V**

**DSTYPE=M**

**DSTYPE=T**

**INSTLN=** `parm list addr`  
*parm list addr*: Rx-type address or register (2) – (12)

**LOG=** `ASIS`  

**LOG=** `NOFAIL`  

**LOG=** `NONE`  

**LOG=** `NOSTAT`  

**OLDVOL=** `old vol addr`  
*old vol addr*: Rx-type address or register (2) – (12)

**APPL=** `applname addr`  
*applname addr*: Rx-type address or register (2) – (12)

**ACCE=** `acee addr`  
*acee addr*: Rx-type address or register (2) – (12)

**ACCLVL=(** `access value` **)**  
*access value addr*: Rx-type address or register (2) – (12)

**ACCLVL=(** `access value` **)**  
*access value addr*: Rx-type address or register (2) – (12)

**RACFIND=YES**

**RACFIND=NO**

**GENERIC=YES**

**GENERIC=ASIS**

**FILESEQ=** `reg`  
*reg*: Register (2) – (12)
The parameters are explained under the standard form of the RACHECK macro with the following exceptions:

- **MF=(E,ctrl addr)** specifies the execute form of the RACHECK macro instruction.
- **RELEASE=(number, CHECK)**, **RELEASE=1.6|1.7|1.8|1.8.1|1.8.2**, **RELEASE=(,CHECK)** specifies the RACF release level of the parameter list to be generated by this macro.

To use the parameters associated with a release, you must specify the number of that release or a later release number. If you specify an earlier release level, the parameter is not accepted by macro processing, and an error message is issued at assembly time. For the parameters that are valid for RELEASE=1.6 and later, see Table 19 on page 303. The default is RELEASE=1.6.

When you specify the RELEASE keyword, checking is done at assembly time. Compatibility between the list and execute forms of the RACHECK macro is validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X’64’ is returned.

### RACINIT: Identify a RACF-defined user

The RACINIT macro is used to provide RACF user identification and verification. The macro instruction identifies a user and verifies that the user is defined to RACF and has supplied a valid password or operator-identification card (OIDCARD parameter), or both.

To issue the RACINIT macro, the calling module must be “authorized,” which means one of the following:

- APF-authorized
- In system key 0–7
RACINIT

- In supervisor state.
Otherwise, the NEWPASS keyword must be omitted and the calling module must
be all of the following:
- In the RACF-authorized caller table
- Fetched from an authorized library
- Reentrant.

Recommendation: If you run programs that issue the RACINIT macro, run those
programs APF-authorized. See z/OS Security Server RACF System Programmer’s
Guide for important information about the authorized caller table.

Note:
1. Only callers in 24-bit addressing mode can issue this macro. Callers executing
   in 31-bit addressing mode who want to use the RACINIT function can code the
   RACROUTE macro.
2. If automatic direction of application updates is active, RACINIT requests are
   not propagated. However, some ICHEINTY requests issued by RACINIT are
   propagated. For more information about which ICHEINTY requests are
   propagated, see “RACROUTE REQUEST=VERIFY: Identify and verify a
RACF-defined user” on page 206.

RACINIT (standard form)
The standard form of the RACINIT macro is written as follows:

```
name

b
RACINIT
b

USERID=userid addr

,PASSWRD=password addr

,START=procname addr

,NEWPASS=new password addr

,GROUP=group addr

,PGMNAME=programmer name addr

,ACTINFO=account addr

,OIDCARD=oid addr

,TERMID=terminal addr
```

userid addr: A-type address or register (2) – (12)
password addr: A-type address or register (2) – (12)
procname addr: A-type address or register (2) – (12)
new password addr: A-type address or register (2) – (12)
group addr: A-type address or register (2) – (12)
programmer name addr: A-type address or register (2) – (12)
account addr: A-type address or register (2) – (12)
oid addr: A-type address or register (2) – (12)
terminal addr: A-type address or register (2) – (12)

Default: zero.
### RACINIT

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>USERID=userid addr</td>
<td>Specifies the user identification of the user who has entered the system. The address points to a 1-byte length field, followed by the user ID, which can be up to 8 characters in length. If the USERID= keyword is omitted, “*” is the default. When verifying a user ID and password from a user, be sure to validate that the user ID and password contain only alphanumeric characters and are 1–8 characters in length. Additionally, the application should fold this information to uppercase.</td>
</tr>
<tr>
<td>PASSWRD=password addr</td>
<td>Specifies the currently defined password of the user who has entered the system. The address points to a 1-byte length field, followed by the password, which can be up to eight characters in length.</td>
</tr>
<tr>
<td>START=procname addr</td>
<td>Specifies the process name of a started task. The address points to an 8-byte</td>
</tr>
</tbody>
</table>
area containing the process name (left-justified and padded with blanks, if necessary). If you do not specify the USERID keyword, but do specify the START keyword, then RACF searches the started-procedure table to determine the user ID.

**NEWPASS=new password addr**
specifies the password that is to replace the user's currently defined password. The address points to a 1-byte length field, followed by the password, which can be up to eight characters in length.

**GROUP=group addr**
specifies the group specified by the user who has entered the system. The address points to a 1-byte length field, followed by the group name, which can be up to eight characters in length.

If the GROUP= keyword is omitted, “*” is the default.

**PGMNAME=programmer name addr**
specifies the address of the name of the user who has entered the system. This 20-byte area is passed to the RACINIT installation exit routine; it is not used by the RACINIT routine.

**ACTINFO=account addr**
specifies the address of a field containing accounting information. This 144-byte area is passed to the RACINIT installation exit routine; it is not used by the RACINIT routine. The accounting field, if supplied, should have the following format:

- The first byte of field contains the number (binary) of accounting fields.
- The following bytes contain accounting fields, where each entry for an accounting field contains a 1-byte length field, followed by the field.

**OIDCARD=oid addr**
specifies the address of the currently defined operator-identification card of the user who has entered the system. The address points to a 1-byte length field, followed by the operator-ID card.

**TERMID=terminal addr**
specifies the address of the identifier for the terminal through which the user is accessing the system. The address points to an 8-byte area containing the terminal identifier. Information specified by TERMID= on an ENVIR=CREATE can be attached to the created ACEE and used in subsequent RACF processing. RACF does not make its own copy of this area when attaching this information to the created ACEE. This area must not be explicitly freed prior to the deletion of the ACEE. For the same reason, the area must reside in a non-task-related storage subpool so that implicit freeing of the area does not occur.

**JOBNAME=jobname addr**
specifies the address of the job name of a background job. The address points to an 8-byte area containing the job name (left-justified and padded with blanks if necessary). The JOBNAME parameter is used by RACINIT during authorization checking to verify the user's authority to submit the job. It is passed to the installation exit routine.

**ENVIR=CREATE**
**ENVIR=CHANGE**
**ENVIR=DELETE**
specifies the action to be performed by the user-initialization component regarding the ACEE:

- CREATE: The user should be verified and an ACEE created.
• **CHANGE**: The ACEE should be modified according to other parameters specified on RACINIT. You can change only the connect group with this option.

• **DELETE**: The ACEE should be deleted. This parameter should be used only if a previous RACINIT has completed successfully.

  **Recommendation**: Issue a `RACINIT,ENVIR=DELETE` to delete only an ACEE that you created. See "Guidelines for changing or deleting an ACEE" on page 315 for alternative options.

`\INSTLN=parm list addr`

specifies the address of an area containing parameter information meaningful to the RACINIT installation exit routine. This area is passed to the installation exit when the exit routine is given control from the RACINIT routine.

The INSTLN parameter can be used by an installation having a user verification or job initiation application, and wanting to pass information from one installation module to the RACINIT installation exit routine.

`\APPL='applname'`

`\APPL=applname addr`

specifies the name of the application issuing the RACINIT. If an address is specified, the address must point to an 8-byte application name, left-justified and padded with blanks if necessary.

`\ACEE=acee addr`

specifies the address of the ACEE.

For ENVIR=DELETE: specifies the address of a fullword that contains the address of the ACEE to be deleted. If ACEE= is not specified, and the TCBSENV field for the task using the RACINIT is nonzero, the ACEE pointed to by the TCBSENV is deleted, and TCBSENV is set to zero. If the TCBSENV and ASXBSENV fields both point to the same ACEE, ASXBSENV is also set to zero. If no ACEE address is passed, and TCBSENV is zero, the ACEE pointed to by ASXBSENV is deleted, and ASXBSENV is set to zero.

For ENVIR=CHANGE: specifies the address of a fullword that contains the address of the ACEE to be changed. If ACEE= is not specified, and the TCBSENV field for the task using the RACINIT is nonzero, the ACEE pointed to by the TCBSENV is changed. If TCBSENV is 0, the ACEE pointed to by ASXBSENV is changed.

For ENVIR=CREATE: specifies the address of a fullword into which the RACINIT function places the address of the ACEE created. If an ACEE is not specified, the address of the newly created ACEE is stored in the TCBSENV field of the task control block. If the ASXBSENV field is set to binary zeros, the new ACEE address is also stored in the ASXBSENV field of the ASXB. If the ASXBSENV field is nonzero, it is not modified. The TCBSENV field is set unconditionally.

**Note:**

1. If you omit USERID, GROUP, and PASSWRD and if you code `ENVIR=CREATE` or if ENVIR=CREATE is used as the default, you receive a return code of X'00' and obtain an ACEE that contains an * (X'5C') in place of the user ID and group name.

2. If ACEE is specified with ENVIR=CREATE, RACF suppresses the creation of a modeling table (MDEL) to reduce the amount of CSA and local system queue area (LSQA) storage required by IMS/VS and CICS/VS installations.
RACINIT

,\texttt{SUBPOOL=\textit{subpool number}}
specifies the storage subpool from which the ACEE and related storage are obtained. The value of subpool can be literally specified or passed through a register. When literally specified, the valid values are 0 through 255. When you use a register, the subpool number is the value of the least significant byte in the register.

,\texttt{SMC=\textit{YES|NO}}
specifies the use of the step-must-complete function of RACINIT processing. SMC=YES specifies that RACINIT processing should continue to make other tasks for the step non-dispatchable. SMC=NO specifies that the step-must-complete function is not used.

\textbf{Note:} SMC=NO should not be used if DADSM ALLOCATE/SCRATCH functions execute simultaneously in the same address space as the RACINIT function.

,\texttt{PASSCHK=\textit{YES|NO}}
specifies whether the user's password or OIDCARD is to be verified. PASSCHK=YES specifies that RACINIT verifies the user's password or OIDCARD. PASSCHK=NO specifies that the user's password or OIDCARD is not verified.

,\texttt{ENCRYPT=\textit{YES|NO}}
specifies whether or not RACINIT encodes the old password, the new password, and the OIDCARD data passed to it.

YES signifies that the data specified by the PASSWRD, NEWPASS, and OIDCARD keywords are not pre-encoded. RACINIT encodes the data before storing it in the user profile or using it to compare against stored data. ENCRYPT=YES is the default for this keyword.

NO signifies that the data specified by the PASSWRD, NEWPASS, and OIDCARD keywords are already encoded. RACINIT bypasses the encoding of this data before storing it in or comparing it against the user profile.

,\texttt{STAT=\textit{ASIS|NO}}
specifies whether the statistics controlled by the installation's options on the RACF SETROPTS command are to be maintained or ignored for this execution of RACINIT. This parameter also controls whether a message is to be issued when the logon is successful.

\textbf{Note:} Messages are always issued if the RACINIT processing is unsuccessful.

If STAT=ASIS is specified or taken by default, the messages and statistics are controlled by the installation's current options on the RACF SETROPTS command.

If STAT=NO is specified, the statistics are not updated. And if the logon is successful, no message is issued.

The default is STAT=ASIS.

,\texttt{LOG=\textit{ASIS|ALL}}
specifies when log records are to be generated.

If LOG=ASIS is specified or defaulted to, only those attempts to create an ACEE that fails generate RACF log records.
If LOG=ALL is specified, any request to create an ACEE, regardless of whether it succeeds or fails, generates a RACF log record. The default is LOG=ASIS.

Parameters for RELEASE=1.6 through 1.8.1

The RELEASE values for which a specific parameter is valid are marked with an 'X'. RELEASE=1.6 is the default.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>RELEASE=1.6</th>
<th>RELEASE=1.7</th>
<th>RELEASE=1.8 or 1.8.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACEE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ACTINFO=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>APPL=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENCRYPT=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENVIR=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>GROUP=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>INSTLN=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>JOBNAME=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LOG=</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>NEWPASS=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>OIDCARD=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PASSCHK=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PASSWRD=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PGMNAME=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RELEASE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SMC=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>START=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>STAT=</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>SUBPOOL=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TERMID=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>USERID=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Guidelines for changing or deleting an ACEE

Delete only an ACEE that you created. Issuing a RACINIT with ENVIR=DELETE specified to delete the existing ACEE can lead to problems if you are not the one who created that environment. The issuer of the ENVIR=CREATE that built the ACEE might have saved a pointer to it and might be expecting it to be available later in processing. Note that this is the case for the initiator’s ACEE. Also, if you delete an ACEE, you might lose tables anchored off that ACEE that are needed later in RACF processing. See z/OS Security Server RACF Diagnosis Guide for overview diagrams of ACEEs and related control blocks that can be useful when diagnosing problems.

Note: When you delete an ACEE that has a third-party ACEE attached, the RACINIT pre- or post-exits get control again for the third-party ACEE as well as for the original ACEE being deleted.
Because RACF services anchor tables off the ACEE and refresh these tables when required, if you make a copy of the ACEE and update fields, avoid passing it to RACHECK or RACDEF. These services anchor tables off the ACEE and refresh these tables when required. If you update fields in a copy, the original ACEE then contains incorrect pointers that result in abends when the original is used or deleted.

If you need to delete or change an ACEE that you did not create, you can use one of the following methods.

- **Save and restore the current ACEE pointers:**
  1. Save the current ACEE pointers from ASXBSENV and TCBSENV.
  2. Clear ASXBSENV and TCBSENV.
  3. Issue RACINIT ENVIR=CREATE.
  4. At the end of processing,
     a. Issue ENVIR=DELETE
     b. Restore ASXBSENV and TCBSENV to the original values.

- **Change the values in the current ACEE:**
  - Issue RACINIT with ENVIR=CHANGE to change the values in the current ACEE.

- **Create, anchor, and delete a third-party ACEE:**
  - Issue RACHECK with USERID= and GROUPID= causing RACF to create, anchor, and delete a third-party ACEE internally.

**Return codes and reason codes**

If the return codes and reason codes you are receiving are not discussed in this macro, refer to the return codes and reason codes described with RACROUTE REQUEST=VERIFY "z/OS V2R1.0 Security Server RACROUTE Macro Reference" on page 232.

When control is returned, register 15 contains one of the following return codes, and register 0 can contain a reason code.

<table>
<thead>
<tr>
<th>Hex Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RACINIT has completed successfully.</td>
</tr>
<tr>
<td>04</td>
<td>The user profile is not defined to RACF.</td>
</tr>
<tr>
<td>08</td>
<td>The password is not authorized.</td>
</tr>
<tr>
<td>0C</td>
<td>The password has expired.</td>
</tr>
<tr>
<td>10</td>
<td>The new password is not valid.</td>
</tr>
<tr>
<td>14</td>
<td>The user is not defined to the group.</td>
</tr>
<tr>
<td>18</td>
<td>RACINIT was failed by the installation exit routine.</td>
</tr>
<tr>
<td>1C</td>
<td>The user's access has been revoked.</td>
</tr>
<tr>
<td>20</td>
<td>RACF is not active.</td>
</tr>
<tr>
<td>24</td>
<td>The user's access to the specified group has been revoked.</td>
</tr>
<tr>
<td>28</td>
<td>OIDCARD parameter is required but not supplied.</td>
</tr>
<tr>
<td>2C</td>
<td>OIDCARD parameter is not valid for specified user.</td>
</tr>
<tr>
<td>30</td>
<td>The user is not authorized to use the terminal. Register 0 contains one of the following reason codes:</td>
</tr>
</tbody>
</table>
Reason Code

Meaning
00    Indicates a normal completion.
04    Indicates the user is not authorized to access the system on this
day, or at this time of day.
08    Indicates the terminal cannot be used on this day, or at this time of
day.
34    The user is not authorized to use the application.
64    Indicates that the CHECK subparameter of the RELEASE keyword was
specified on the execute form of the RACINIT macro; however, the list
form of the macro does not have the proper RELEASE parameter. Macro
processing terminates.

Example 1
Use the standard form of the macro to do the following tasks:
• Create an ACEE for the user ID and its default group.
• Verify that the user named USERNAME is a valid user.
• Verify that the password called PASSWORD is valid.
RACINIT ENVIR=CREATE,USERID=USERNAME,PASSWRD=PASSWORD

Example 2
Use the standard form to do the following tasks:
• Verify that the user named USERNAME is a valid user.
• Verify that the group named GROUPNAM is a valid group.
• Verify that USERNAME is defined to the group.
• Create an ACEE for the user and group and put its address in ACEEANCH.
• Specify that the user’s password is not required.
RACINIT ENVIR=CREATE,USERID=USERNAME,GROUP=GROUPNAM,ACEE=ACEEANCH,PASSCHK=NO

Example 3
Use the standard form of the macro to delete the ACEE of the current task or
address space, or both.
RACINIT ENVIR=DELETE

RACINIT (list form)

The list form of the RACINIT macro is written as follows:

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RACINIT.</td>
</tr>
</tbody>
</table>

RACINIT

<table>
<thead>
<tr>
<th>b</th>
<th>One or more blanks must follow RACINIT.</th>
</tr>
</thead>
</table>

| USERID=userid addr | userid addr: A-type address. |
| PASSWRD=password   | password addr: A-type address. |
ADDR

.START=procname addr
.START=procname addr: A-type address.

.NEWPASS=new password addr
.NEWPASS=new password addr: A-type address.

.GROUP=group addr
.GROUP=group addr: A-type address.

.PGMNAME=programmer name addr
.PGMNAME=programmer name addr: A-type address.

.ACTINFO=account addr
.ACTINFO=account addr: A-type address.

.OIDCARD=oid addr
.OIDCARD=oid addr: A-type address.

.TERMID=terminal addr
.TERMID=terminal addr: A-type address.

.JOBNAME=jobname addr
.JOBNAME=jobname addr: A-type address.

.ENVIR=CREATE
.ENVIR=CREATE

.ENVIR=CHANGE
.ENVIR=CHANGE

.ENVIR=DELETE
.ENVIR=DELETE

.INSTLN=parm list addr
.INSTLN=parm list addr: A-type address.

.APL='applname'
.APL='applname'

.APL=applname addr
.APL=applname addr: A-type address.

.ACEE=acee addr
.ACEE=acee addr: A-type address.

.SUBPOOL=subpool number
.SUBPOOL=subpool number: Decimal digit 0–255.

.SMC=YES
.SMC=YES

.SMC=NO
.SMC=NO

.PASSCHK=YES
.PASSCHK=YES

.PASSCHK=NO
.PASSCHK=NO

.ENCRYPT=YES
.ENCRYPT=YES

.ENCRYPT=NO
.ENCRYPT=NO

.STAT=ASIS
.STAT=ASIS

.STAT=NO
.STAT=NO

.LOG=ASIS
.LOG=ASIS

.LOG=ALL
.LOG=ALL

.MF=L

RACINIT

addr

 Định nghĩa các tham số:

- **.START=procname addr**: A-type address.
- **.NEWPASS=new password addr**: A-type address.
- **.GROUP=group addr**: A-type address.
- **.PGMNAME=programmer name addr**: A-type address.
- **.ACTINFO=account addr**: A-type address.
- **.OIDCARD=oid addr**: A-type address.
- **.TERMID=terminal addr**: A-type address.
- **.JOBNAME=jobname addr**: A-type address.

**.ENVIR=CREATE**

Default: ENVIR=CREATE

Note:

1. ENVIR=CHANGE cannot be specified with USERID=,
   PASSWRD=, START=, NEWPASS=, ACTINFO=, PGMNAME=,
   OIDCARD=, or TERMID= parameters.
2. ENVIR=DELETE cannot be specified with APPL=, USERID=,
   PASSWRD=, START=, NEWPASS=, GROUP=, ACTINFO=,
   PGMNAME=, OIDCARD=, or TERMID= parameters.

**.ENVIR=DELETE**

**.INSTLN=parm list addr**: A-type address.

**.APPL='applname'**

**.APPL=applname addr**: A-type address.

**.ACEE=acee addr**: A-type address.

**.SUBPOOL=subpool number**: Decimal digit 0–255.

**.SMC=YES**

Default: SMC=YES

**.SMC=NO**

Default: SMC=NO

**.PASSCHK=YES**

Default: PASSCHK=YES

**.PASSCHK=NO**

**.ENCRYPT=YES**

Default: ENCRYPT=YES

**.ENCRYPT=NO**

**.STAT=ASIS**

Default: STAT=ASIS

**.STAT=NO**

Default: STAT=NO

**.LOG=ASIS**

Default: LOG=ASIS

**.LOG=ALL**

Default: LOG=ALL

**.MF=L**
The parameters are explained under the standard form of the RACINIT macro instruction with the following exception:

\[ ,MF=L \]

specifies the list form of the RACINIT macro instruction.

## RACINIT (execute form)

The execute form of the RACINIT macro is written as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>USERID=</td>
<td>userid addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,PASSWRD=</td>
<td>password addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,START=</td>
<td>procname addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,NEWPASS=</td>
<td>new password addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,GROUP=</td>
<td>group addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,PGMNAME=</td>
<td>programmer name addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,ACTINFO=</td>
<td>account addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,OIDCARD=</td>
<td>oid addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,TERMID=</td>
<td>terminal addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,JOBNAME=</td>
<td>jobname addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,INSTLN=</td>
<td>parm list addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,APPL=</td>
<td>applname addr: Rx-type address or register (2) – (12)</td>
</tr>
<tr>
<td>,ACEE=</td>
<td>acee addr: Rx-type address or register (2) – (12)</td>
</tr>
</tbody>
</table>

**Note:**

1. ENVIR=CHANGE cannot be specified with USERID=, PASSWRD=, START=, NEWPASS=, ACTINFO=, PGMNAME=, OIDCARD=, or TERMID= parameters.
2. ENVIR=DELETE cannot be specified with APPL=, USERID=, PASSWRD=, START=, NEWPASS=, GROUP=, ACTINFO=, PGMNAME=, OIDCARD=, or TERMID= parameters.
The parameters are explained under the standard form of the RACINIT macro instruction with the following exceptions:

,.MF=(E,ctrl addr)  
ctrl addr: Rx-type address or register (1) or (2) – (12)

,.RELEASE=(number,CHECK)  
number: See Standard Form  
Default: RELEASE=1.6

To use the parameters associated with a release, you must specify the release number of that release or a later release number. If you specify an earlier release level, the parameter is not accepted by macro processing, and an error message is issued at assembly time. If you specify a parameter with an incompatible release level, the parameter is not accepted by macro processing. An error message is issued at assembly time. For the parameters that are valid for RELEASE=1.6 and later, see Table 20 on page 315.

The default is RELEASE=1.6.

When you specify the RELEASE keyword, checking is done at assembly time. Compatibility between the list and execute forms of the RACINIT macro is validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.
**RACLIST: Build in-storage profiles**

RACLIST is used to build in-storage profiles for RACF-defined resources. RACLIST processes only those resources described by class descriptors. The primary advantage of using the RACLIST macro is to use the resource-grouping function and to improve resource-authorization-checking performance.

The module calling the RACLIST macro must either be authorized (APF-authorized, in system key 0–7, or in supervisor state) or reentrant in the RACF-authorized caller table and fetched from an authorized library.

**Recommendation:** If you run programs that issue the RACLIST macro, run those programs APF-authorized. See [z/OS Security Server RACF System Programmer’s Guide](#) for important information about the authorized caller table.

**Note:** Only callers in 24-bit addressing mode can issue this macro. Callers executing in 31-bit addressing mode who want to use the RACLIST function can code the RACROUTE macro.

---

**RACLIST (standard form)**

The standard form of the RACLIST macro is written as follows:

```
___________________________________________________________________________________
name name

\| name: Symbol. Begin name in column 1.

b

One or more blanks must precede RACLIST.

RACLIST

b

One or more blanks must follow RACLIST.

___________________________________________________________________________________
CLASS='class name'
CLASS=class name addr

\| class name: 1–8 character class name

\| class name addr: A-type address or register (2) – (12)

LIST=list addr

\| list addr: A-type address or register (2) – (12)

ACEE=acee addr

\| acee addr: A-type address or register (2) – (12)

INSTLN=parm list addr

\| parm list addr: A-type address or register (2) – (12)

APPL='applname'

\| applname: 1–8 character name

\| applname addr: A-type address or register (2) – (12)

SUBPOOL=(sub#1,sub#2)

\| sub#1,sub#2: Decimal digit 0–255.

ENVIR=CREATE

Default: ENVIR=CREATE

ENVIR=DELETE

\| OWNER=YES

\| Default: OWNER=NO

Owner=NO
```

The parameters are explained as follows:

**CLASS='class name'**
RACLIST

CLASS=class name addr
specifies that RACLIST is to build an in-storage profile for the resources of the specified class. If an address is specified, the address must point to an 8-byte field containing the class name, left-justified and padded with blanks if necessary.

The class name must be a valid, active class as defined in the class descriptor table. It must also be a member class. If the member class has a grouping class associated with it, RACF utilizes both the member and the grouping class when building the in-storage profiles.

LIST=addr
specifies the address of a list of resource names for which RACLIST is to build the in-storage profiles. The list consists of a 2-byte field containing the number of the names in the list, followed by one or more variable-length names. Each name consists of a 1-byte length field, which is the length of the name, followed by the name. A zero in the 2-byte field causes the operand to be omitted. If LIST= is omitted, in-storage profiles are built for all the profiles defined to RACF in the given class as well as each member for a resource grouping associated with the specified class.

Note: This operand can be specified only with ENVIR=CREATE. If ENVIR=DELETE is specified, the RACLIST macro issues a return code of 18.

ACEE=acee addr
specifies the address of the ACEE. The ACEE points to the in-storage profiles. If an ACEE is not specified, RACF uses the TASK ACEE pointer in the extended TCB called the TCBSENV. Otherwise, or if the TASK ACEE pointer is zero, RACF uses the main ACEE to obtain the list of the in-storage profiles. The main ACEE is pointed to by the ASXBSENV field of the address-space extension block. If an ACEE is not specified and there is no main ACEE, the in-storage profiles are not constructed.

INSTLN=parm list addr
specifies the address of an area that contains parameter information for the RACLIST installation exit. The address is passed to the installation exit when the RACLIST routine gives control to the exit. An application or an installation program can use the INSTLN parameter to pass information to the RACLIST installation exit.

APPL='applname'

APPL=applname addr
specifies the name of the application requesting the authorization checking. This information is not used for the authorization-checking process but is made available to the installation exit or exits. If an address is specified, it should point to an 8-byte area containing the application name, left justified and padded with blanks if necessary.

SUBPOOL=(sub#1,sub#2)
 specifies the subpool numbers of the storage into which the components of the in-storage profiles are to be built. sub#1 represents the subpool of the profile index. sub#2 represents the subpool of the profile proper. If the subpools are not specified they default to subpool 255. Registers can be used to specify sub#1 and sub#2. In that case, the least significant byte in the register is the subpool value.

ENVIR=CREATE
ENVIR=DELETE
specifies the action to be performed by the RACLIST macro.
CREATE: In-storage profiles for the specified class are to be built. The RACLIST function issues a return code of 18, if an in-storage list currently exists for the specified class.

DELETE: The in-storage profiles for the specified class are to be freed. If class is not specified, the in-storage profiles for all classes are freed.

**Note:** It is the responsibility of the user issuing the RACLIST macro to assure that no multitasking that results in the issuing of a RACHECK, FRACHECK, RACINIT, or RACLIST macro occurs at the same time that the RACLIST occurs.

\[ \text{OWNER=YES} \]
\[ \text{OWNER=NO} \]

specifies that the resource owner is to be placed in the profile access list with the ALTER authority. If the OWNER= operand is omitted, the default is NO.

### Parameters for RELEASE=1.6 through 1.8.1

The RELEASE values for which a specific parameter is valid are marked with an ‘X’. RELEASE=1.6 is the default.

**Table 21. RACLIST parameters for RELEASE=1.6 through 1.8.1**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>RELEASE=1.6</th>
<th>RELEASE=1.7</th>
<th>RELEASE=1.8 or 1.8.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACEE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>APPL=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CLASS=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENVIR=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>INSTLN=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LIST=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>OWNER=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RELEASE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SUBPOOL=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

### Return codes and reason codes

If the return codes and reason codes you are receiving are not discussed in this macro, refer to the return codes and reason codes described in RACROUTE REQUEST=LIST “Return codes and reason codes” on page 155.

When control is returned, register 15 contains one of the following return codes, and register 0 can contain a reason code.

**Hex code**

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
</tr>
</tbody>
</table>

**Reason code**

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
</tr>
<tr>
<td>4</td>
</tr>
</tbody>
</table>
04 Unable to perform the requested function. Register 0 contains additional codes as follows:

<table>
<thead>
<tr>
<th>Reason code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Unable to establish an ESTAE environment.</td>
</tr>
<tr>
<td>1</td>
<td>The function code (the third byte of the parameter list) does not represent a valid function. 01 represents the RACF manager; 02 represents the RAclist macro.</td>
</tr>
</tbody>
</table>

08 The specified class is not defined to RACF.
0C An error was encountered during RAclist processing.
10 RACF or the resource class is not active, or both.
14 RAclist installation exit error occurred.
18 Parameter-list error.
1C RACF is not installed, or an insufficient level of RACF is installed.
64 Indicates that the CHECK subparameter of the RELEASE keyword was specified on the execute form of the RAclist macro; however, the list form of the macro does not have the proper RELEASE parameter. Macro processing terminates.

Note: If the resource class specified by the CLASS= operand is inactive, RAclist does not build the in-storage profiles and a code of 0C is returned. If the resource-group class is not active, RAclist builds an in-storage profile but only from the individual resource profiles; resource-group profiles are ignored.

**Example 1**
Use the standard form of the macro to build in-storage profiles for all the profiles in the DASDVOL class and chain them off the ACEE whose address is pointed to by ACEEADDR.

```
RAclist CLASS='DASDVOL',ACEE=ACEEADDR,ENVIR=CREATE
```

**Example 2**
Use the standard form of the macro to build in-storage profiles for all the profiles whose names are in a list named PROFLIST and DASDVOL class. Chain them from the task ACEE or address space ACEE.

```
RACLST CLASS='DASDVOL',LIST=PROFLIST,ENVIR=CREATE
```

```
PROFLIST DS 0CL35
PROFNUM DC XL2'0005'
PROF1 DC AL1(6),CL6'DASD01'
PROF2 DC AL1(6),CL6'DASD02'
PROF3 DC AL1(5),CL5'DASDA'
PROF4 DC AL1(5),CL5'DASDB'
PROF5 DC AL1(6),CL6'MYDASD'
```

**Example 3**
Use the standard form of the macro to delete the in-storage profiles for the DASDVOL class.

```
RAclist CLASS=DASDVOL,ENVIR=DELETE
```
**RACLIST (list form)**

The list form of the RA CLIST macro is written as follows:

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RA CLIST.</td>
</tr>
</tbody>
</table>

RA CLIST

| b                    | One or more blanks must follow RA CLIST. |

CLASS=''class name'   | class name: 1–8 character class name |
CLASS='class name addr'| class name addr: A-type address       |

LIST='list addr'      | list addr: A-type address             |
ACCE='acct addr'      | acct addr: A-type address.            |
INSTLN='parm list addr'| parm list addr: A-type address.        |

APPL='applname'       | applname: 1–8 character name          |
APPL='applname addr'  | applname addr: A-type address         |

SUBPOOL=(sub#1,sub#2) | sub#1,sub#2: Decimal digit 0–255.     |
| Default: 255.        |                                        |

ENVIR=CREATE         | Default: ENVIR=CREATE                 |
ENVIR=DELETE         |                                        |

OWNER=YES           | Default: OWNER=NO                     |
OWNER=NO            |                                        |

MF=L

The parameters are explained under the standard form of the RA CLIST macro with the following exception:

```
name
```

specifies the list form of the RA CLIST macro instruction.

**RACLIST (execute form)**

The execute form of the RA CLIST macro is written as follows:

<table>
<thead>
<tr>
<th>name</th>
<th>name: Symbol. Begin name in column 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>One or more blanks must precede RA CLIST.</td>
</tr>
</tbody>
</table>

RA CLIST

| b                    | One or more blanks must follow RA CLIST. |
The parameters are explained under the standard form of the RACLIST macro with the following exceptions:

\[ \text{MF}=(E,\text{ctrl addr}) \]

specifies the execute form of the RACLIST macro instruction, using a remote, control-program parameter list.

\[ \text{RELEASE}=\text{number} \]
\[ \text{RELEASE}=(,\text{CHECK}) \]
\[ \text{RELEASE}=(\text{number},\text{CHECK}) \]

specifies the RACF release level of the parameter list to be generated by this macro.

To use the parameters associated with a release, you must specify the number of that release or a later release number. If you specify an earlier release level, the parameter is not accepted by macro processing, and an error is issued at assembly time. For the parameters that are valid for RELEASE=1.6 and later, see Table 21 on page 323.

The default is RELEASE=1.6. When you specify the RELEASE keyword, checking is done at assembly time. Compatibility between the list and execute forms of the RACLIST macro is validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.
RACSTAT macro

The RACSTAT macro is used to determine whether RACF is active, and, optionally, determine whether RACF protection is in effect for a given resource class. The RACSTAT macro can also be used to determine whether a resource-class name is defined to RACF.

RACSTAT is a branch-entered service that uses standard linkage conventions.

Restriction: Class names in the CDT class are not processed by the RACSTAT macro.

Note: For RACF release 1.6 and prior releases, only callers in 24-bit addressing mode can issue this macro.

RACSTAT macro (standard form)

The standard form of the RACSTAT macro is written as follows:

```
___________________________________________________________________________________
name name: Symbol. Begin name in column 1.
/bslash
One or more blanks must precede RACSTAT.
RACSTAT
/bslash
One or more blanks must follow RACSTAT.
___________________________________________________________________________________
/ ,CLASS=class name
   class name: 1–8 character class name
,CLASS=class name addr
   class name addr: A-type address or register (2) – (12)
,ENTRY=entry addr
   entry addr: A-type address or register (2) – (12)
___________________________________________________________________________________
```

The parameters are explained as follows:

- `CLASS='class name'` specifies the class name for which RACF authorization checking is performed. The name can be explicitly defined on the macro by enclosing the name in quotes. If specified, the address must point to an 8-byte field containing the class name, left-justified and padded with blanks if necessary. If `CLASS=` is omitted, the status of RACF is returned.

The class name specified must be a general resource defined to RACF in the class descriptor table (CDT). For more information, see Appendix E, “Supplied class descriptor table entries,” on page 397.

Note: The classes DATASET, USER, and GROUP are not in the class descriptor table.

Restriction: Class names in the CDT class are not processed by the RACSTAT macro.
RACSTAT

ENTRY=entry addr

specifies the address of a 4-byte area that is set to the address of the specified
class in the class descriptor table. This operand is ignored when the CLASS=
operand is omitted.

Parameters for RELEASE=1.6 through 1.8.1

The RELEASE values for which a specific parameter is valid are marked with an
‘X’. RELEASE=1.6 is the default.

Table 22. RACSTAT parameters for RELEASE=1.6 through 1.8.1

<table>
<thead>
<tr>
<th>Parameter</th>
<th>RELEASE=1.6</th>
<th>RELEASE=1.7</th>
<th>RELEASE=1.8 or 1.8.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLASS=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENTRY=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>RELEASE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Return codes

If the return codes and reason codes you are receiving are not discussed in this
macro, refer to the return codes and reason codes described with RACROUTE
REQUEST=STAT "Return codes and reason codes" on page 181.

When control is returned, register 15 contains one of the following return codes:

Hex Code   | Meaning
-----------|----------------------------------------------------
00          | RACF is active and, if CLASS= was specified, the class is active.
04          | RACF is active; the class is inactive.
08          | RACF is active; the class is not defined to RACF.
0C          | RACF is inactive and, if CLASS= was specified, the class is active.
10          | RACF is inactive; the class is inactive.
14          | RACF is inactive; the class is not defined to RACF.
18          | RACF is not installed or an insufficient level of RACF is installed.
64          | Indicates that the CHECK subparameter of the RELEASE keyword was
            | specified on the execute form of the RACSTAT macro; however, the list
            | form of the macro does not have the proper RELEASE parameter. Macro
            | processing terminates.

Note: The class descriptor entry for the specified class is returned to the caller (in
the 4-byte area addressed by the entry address), for return codes 00, 04, 0C, and 10.

Example 1

Determine whether the DASDVOL class is active and retrieve the address of its
class descriptor. A fullword, CDADDR, contains the class descriptor address.
RACSTAT CLASS=’DASDVOL’,ENTRY=CDADDR

RACSTAT (list form)

The list form of the RACSTAT macro instruction is written as follows:

name name Symbol. Begin name in column 1.
The parameters are explained under the standard form of the RACSTAT macro with the following exception:

**MF=L**

specifies the list form of the RACSTAT macro.

### RACSTAT (execute form)

The execute form of the RACSTAT macro is written as follows:

**name**

name: Symbol. Begin name in column 1.

One or more blanks must precede RACSTAT.

**RACSTAT**

One or more blanks must follow RACSTAT.

The parameters are explained under the standard form of the RACSTAT macro with the following exceptions:

**MF=(E, ctrl addr)**

specifies the execute form of the RACSTAT macro, using a remote, control-program parameter list.

**RELEASE=number**

**RELEASE=(, CHECK)**

**RELEASE=(number, CHECK)**
RACSTAT

\texttt{,RELEASE=(number,CHECK)}
specifies the RACF release level of the parameter list to be generated by this macro.

Certain parameters can be specified only with particular releases. If you specify a parameter with an incompatible release level, the parameter is not accepted by the macro processing. An error message is issued at assembly time. For the parameters that are valid for RELEASE=1.6 and later, see Table 22 on page 328. When you specify the RELEASE keyword, checking is done at assembly time.

Compatibility between the list and execute forms of the RACSTAT macro is validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X'64' is returned.

The default is RELEASE=1.6.

RACXTRT macro (standard form)

The RACXTRT macro is used to retrieve or replace certain specified fields from a RACF profile or to encode certain clear-text (readable) data.

This macro is only available to authorized callers (APF-authorized, in system key 0-7, or in supervisor state).

When activated, automatic direction of application updates propagates RACXTRT TYPE=REPLACE requests on to selected remote nodes. Only RACXTRT TYPE=REPLACE requests with return code 0 are propagated.

Note:
1. Encoding, replacement, and extraction are mutually exclusive.
2. Only callers in 24-bit addressing mode can issue this macro. Callers executing in 31-bit addressing mode who want to use the RACXTRT function can code the RACROUTE macro.

The following RACXTRT functions are Programming Interfaces:
- Retrieving or updating fields in any other product segment (including WORKATTR) in the user, group and resource profiles.
- Retrieving or updating the following installation-reserved fields:
  - USERDATA
  - USRCNT
  - USRDATA
  - USRFLG
  - USRNM
- Retrieving the current or a specified user's default group or password.

The following RACXTRT function is part of the Programming Interface but is not recommended for use because the macro is not going to be enhanced in future releases.
- Retrieving or updating fields in the base segment of a user, resource, or group profile.
The standard form of the RACXTRT macro is written as follows:

```
name

One or more blanks must precede RACXTRT.

RACXTRT

One or more blanks must follow RACXTRT.
```

<table>
<thead>
<tr>
<th>TYPE</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE=EXTRACT</td>
<td>A-type address, or register (2) – (12)</td>
</tr>
<tr>
<td>TYPE=EXTRACTN</td>
<td>A-type address, or register (2) – (12)</td>
</tr>
<tr>
<td>TYPE=REPLACE</td>
<td>A-type address, or register (2) – (12)</td>
</tr>
<tr>
<td>TYPE=ENCRYPT</td>
<td>A-type address, or register (2) – (12)</td>
</tr>
</tbody>
</table>

If TYPE=EXTRACT or EXTRACTN is specified:

```
,ENTITY=profile name

profile name addr: A-type address, or register (2) – (12)

,ACEE=acee-address

acee: A-type address, or register (2) – (12)

,VOLSER=volser-address

vol address: A-type address, or register (2) – (12)

,G generic=ASIS

Default: ASIS

,G generic=YES

Default: NO

,FLDACC=YES

Default: NO

,FLDACC=NO

```

If TYPE=REPLACE is specified:

```
,CLASS=class name

class name: 1–8 character class name

,CLASS=class name addr

class name addr: A-type address or register (2) – (12)

,SEGMENT=segment name

segment name: 1–8 character name

,SEGMENT=segment name addr

segment name addr: A-type address or register (2) – (12)

,FIELDS=field addr

field addr: A-type address or register (2) – (12)
```
If TYPE=ENCRYPT is specified:

\[\text{ENCRYPT=(data address,DES)}\]
\[\text{ENCRYPT=(data address,HASH)}\]
\[\text{ENCRYPT=(data address,INST)}\]

Note: If TYPE=ENCRYPT is specified, the only other allowable parameters are ENTITY, RELEASE, ENCRYPT, with ENCRYPT being required.

The parameters are explained as follows:

**TYPE=EXTRACT**

specifies the function to be performed by the extract function routine.

With Release 1.8 and later, RACXTRT can provide additional function: it can extract information from any field in any profile. See Appendix B, “RACF database templates,” on page 347 for a definition of the type and name of each field in each profile. If you specify EXTRACT, the macro extracts information from the profile determined by the ENTITY and CLASS keywords. Specifically, RACF extracts the fields specified in the FIELDS keyword from the segment specified by the SEGMENT keyword. If you do not specify ENTITY, RACF retrieves the desired information from the current user's profile.

To use TYPE=EXTRACT to extract field information from a profile, you must specify RELEASE=1.8 or later.

Note: If you specify TYPE=EXTRACT, do not specify ENCRYPT.

Upon return, register 1 contains the address of a result area that begins with a fullword containing the area's subpool number and length. It is your responsibility to issue a FREEMAIN to release the area after you are through using it.

The fields in the result area are in the order below:

<table>
<thead>
<tr>
<th>Offset (Dec)</th>
<th>Data</th>
<th>Length (Dec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Subpool of area</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>Length of area</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>Offset to start of optional field to contain segment data</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>Flag</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>Reserved</td>
<td>17</td>
</tr>
<tr>
<td>24</td>
<td>Specified or current user's user ID, if CLASS=USER</td>
<td>8</td>
</tr>
<tr>
<td>32</td>
<td>Specified user's default connect group or current user's current connect group, if CLASS=USER</td>
<td>8</td>
</tr>
</tbody>
</table>

In general, RACF returns field data in the order it was specified, with a 4-byte length field preceding each profile field. For example, if you are extracting a single field, you receive a 4-byte length field that contains the length of the field that
follows. If the requested field is a variable length field, there is no additional length byte.

| 4 bytes (length of data) | data |

If you are extracting a combination field (representing one or more fields), you receive:
- A 4-byte length field that contains the combined length of all the fields that follow
- A combination field made up of 4-byte length fields followed by their respective individual data fields.

<table>
<thead>
<tr>
<th>Total length of combination field</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 bytes (length of data1)</td>
</tr>
<tr>
<td>4 bytes (length of data2)</td>
</tr>
</tbody>
</table>

If you are extracting a single field within a repeat group, you receive:
- A 4-byte length field that contains the combined length of all the fields that follow.
- A 4-byte length field that indicates the length of the specified field in the first occurrence of the repeat group. This is followed by a 4-byte length field that indicates the length of the specified field in the second occurrence of the repeat group. The pattern repeats until all the occurrences of the repeat group are accounted for.

<table>
<thead>
<tr>
<th>Total length of all the following fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 bytes (length of data1)</td>
</tr>
<tr>
<td>4 bytes (length of data1)</td>
</tr>
</tbody>
</table>

If you are extracting a combination field (representing one or more fields) within a repeat group, you receive:
- A 4-byte length field that contains the combined length of all the fields that follow.
- A combination field consisting of a 4-byte length field indicating the length of the individual data field that follows it, followed by the next 4-byte length field indicating the length of the next individual data field. The pattern repeats until all the individual fields that make up the combination field are accounted for. At the next occurrence of the repeat group the pattern begins again.
Specifying the name of a repeat-group count field retrieves only the 4-byte length followed by the 4-byte repeat group count.

When a field to be extracted is empty, the following results:

- For fixed-length fields, RACF returns the default as specified by the template definitions. The default for flag fields is X’00’. The default for fixed-length fields in the base segment of the profile in binary ones. The default for fixed-length fields in other segments is binary zeros.
- For variable-length fields, RACF returns a length of zero and no data.

If CLASS=USER, when you specify EXTRACT, the macro extracts the user ID, connect group and, optionally, the encoded password from the user profile.

**TYPE=EXTRACTN**

specifies the function to be performed by the EXTRACT function routine.

**Note:** If you specify TYPE=EXTRACTN, do not specify ENCRYPT=.

Upon return, register 1 contains the address of a result area that begins with a fullword containing the area’s subpool number and length. To see the format of the result area, see the explanation of TYPE=EXTRACT, above. At offset 6 in the result area, there is a flag. If the flag has a X’80’, the name returned is generic.

If you specify EXTRACTN, the macro extracts information from the profile that follows the profile determined by the ENTITY and CLASS keywords. From that next profile, RACF extracts the fields specified in the FIELDS keyword from the segment specified by the SEGMENT keyword. In addition, RACF returns the name of the profile from which it extracted the data.

**TYPE=REPLACE**

specifies the function to be performed by the EXTRACT function routine.

**Note:** If you specify TYPE=REPLACE, do not specify ENCRYPT=.

Use of the REPLACE option to update a profile requires a thorough knowledge of the interrelationships of fields within a profile and of the potential relationships between profiles. For instance, if you use RACXTRT to update a password, you should also update the password change date and password history information.

If you specify TYPE=REPLACE, RACF takes the information in the fields specified in the FIELDS parameter and pointed to by SEGDATA, and places that information in the designated SEGMENT. (The SEGMENT is within the profile determined by the ENTITY and CLASS keywords.) If you specify TYPE=REPLACE, you must specify FIELDS, SEGDATA=, and RELEASE=1.8 or later. If you want to replace a segment other than the base segment, you must specify the SEGMENT keyword with the segment you want. If you do not specify SEGMENT, the segment defaults to the base segment.
With 1.8 and later, if you want to create a TSO segment, you can do so by specifying the RACXTRT macro in the following way:

```
TYPE=REPLACE   SEGMENT=TSO
```

`,SUBPOOL=subpool number`

specifies the storage subpool from which the extract-function routine obtains an area needed for the extraction. If this parameter is not specified, it defaults to 229.

**Note:** Care should be taken in selecting a subpool. Selecting a fetch-protected subpool or subpool 0 might result in programs being unable to access or free retrieved data.

`,DERIVE=YES`

specifies that the desired field is obtained from the DFP segment of the appropriate profile. To specify DERIVE, you must also specify RELEASE=1.8.1.

DERIVE requests are limited to the DFP segment of the data set and user profiles. The following information is an explanation of the DERIVE processing for both DATASET and USER requests.

- **DATASET**
  
  Specifying the DERIVE=YES keyword with CLASS=DATASET and FIELDS=RESOWNER causes RACF to perform additional processing, other than simply extracting the data set resource owner from the data set profile. DFP uses this retrieved information for authority checking when allocating a new data set.

  To process the request, RACF first attempts to extract the RESOWNER field from the DATASET profile specified by the ENTITY keyword. If the profile exists and the RESOWNER field contains data, RACF checks to see whether that data is the user ID of a USER or GROUP currently defined to RACF. If so, RACF returns that user ID along with a reason code that indicates whether the user ID is that of a USER or GROUP.

  If RACF does not find a profile that matches the data set name specified by the ENTITY keyword, RACF attempts to locate the generic data set profile that protects that data set name.

  If it finds the generic profile, and the RESOWNER field contains data, RACF checks to see whether that data is the user ID of a USER or GROUP currently defined to RACF. If so, RACF returns that user ID along with a reason code that indicates whether the user ID is that of a USER or GROUP.

  If RACF does not find a generic profile or the retrieved data is neither a USER nor a GROUP, RACF returns the high-level qualifier from the name specified on the ENTITY keyword, along with a reason code that indicates whether that high-level qualifier matches a defined USER or GROUP, or neither.

  You specify a DERIVE request for RESOWNER as follows:

  ```
  RACROUTE REQUEST=EXTRACT, TYPE=EXTRACT,
  ENTITY=DSNAME,
  VOLSER=MYDASD,
  CLASS='DATASET',
  FIELDS=RESFLDS,
  SEGMENT='DFP',
  DERIVE=YES,
  RELEASE=1.8.1
  ```
The purpose of specifying the DERIVE=YES keyword with CLASS=USER is to obtain the desired DFP-field information (STORCLAS or MGMTCLAS) from the profile of the user. If the user’s profile does not contain the desired DFP fields, RACF goes to the user's default group and attempts to obtain the information for the remaining fields from the GROUP profile (the remaining fields being those that do not contain information in the USER profile).

You specify a DERIVE request for information from a USER profile as follows:

```
RACROUTE REQUEST=EXTRACT,TYPE=EXTRACT,
ENTITY=USER01,
CLASS='USER',
FIELDS=STRFLDS,
SEGMENT='DFP',
DERIVE=YES,
RELEASE=1.8.1
```

RACF processes the DERIVE keyword only if it is specified with the DATASET or USER class. In addition, for DERIVE processing to occur, SEGMENT=DFP and RELEASE=1.8.1 must also be specified.

### FIELDS=address

Specifies the address of a variable-length list. The first field is a 4-byte field that contains the number of profile-field names in the list that follows. Each profile-field name is 8 bytes long, left-justified, and padded to the right with blanks. The allowable field names for each type of profile are in the template listings in Appendix B, “RACF database templates,” on page 347. To see how to specify the FIELDS keyword, see the TYPE=REPLACE example that follows.

- If you specify RELEASE=1.6 or later, or allow the keyword to default, the following options exist:
  - The only acceptable value of the count field is 1.
  - The only acceptable field name is PASSWORD. Use this parameter when you want to extract the user’s encoded password in addition to the user ID and connect group. RACF returns the encoded password in the result area at an offset from the start of the area specified by the halfword at offset 4. (See the result area under TYPE=EXTRACT.)
- If you specify RELEASE=1.8 or later, the following options exist:
  - The count field can contain numbers from 1 through 255.
  - The field names can be any of the field names in the template listings.
ENTITY= parameters, and returns the contents in the result area. (See result area explained under the EXTRACT keyword.)

With Release 1.8, you can specify TYPE=REPLACE. RACF replaces or creates the indicated fields in the profile specified on the CLASS and ENTITY keywords with the data pointed to by the SEGDATA keyword.

**Note:**
1. Do not replace a repeat group count field. Doing so causes unpredictable results.
2. You cannot replace an entire repeat group, a single occurrence of a repeat group, or a single existing field in a repeat group. If you attempt to do so, RACF adds the data to the existing repeat group or groups.
   The only things you can do is retrieve all occurrences of specified fields within a repeat group or add a new occurrence of a repeat group.
3. If you add occurrences of a repeat group, RACF places those additions at the beginning (front) of the repeat group.

The following example of TYPE=REPLACE replaces fields in the base segment. It shows one way to code the macro and the declarations necessary to make the macro work.

```
RACXTRT TYPE=REPLACE,
  CLASS='USER',
  ENTITY=USERID,
  FIELDS=FLDLIST,
  SEGDATA=SEGDLIST,
  SEGMENT=BASE
```

```
USERID DC CL8,'BILL'
FLDLIST DC A(3)
  DC CL8'AUTHOR'
  DC CL8'DFLTGRP'
  DC CL8'NAME'
SEGDLIST DC AL4(6),CL6'JSMITH'
  DC AL4(8),CL8'SECURITY'
  DC AL4(11),CL11'BILL THOMAS'
BASE DC CL8'BASE'
```

When the replacement action takes place, the following results occur:
- JSMITH is placed in the AUTHOR field in the profile.
- SECURITY is placed in the DFLTGRP field in the profile.
- BILL THOMAS is placed in the NAME field in the profile.

The following example of TYPE=EXTRACT retrieves the universal access from a fully-qualified generic data set profile. The information is retrieved in a work area created in SUBPOOL 1.

```
RACXTRT TYPE=EXTRACT,
  CLASS='DATASET',
  ENTITY=DSN,
  FIELDS=FLDS,
  GENERIC=YES,
  SUBPOOL=1
  RELEASE=1.8,
  SEGMENT='TSO'
```

```
DSN DC CL44'SYS1.LINKLIB'
FLDS DC A(1)
  DC CL8 'UACC'
```
 TYPE=ENCRYPT
specifies the function to be performed by the extract-function routine.

If TYPE=ENCRYPT is specified, the operation performed is data encoding. The ENCRYPT keyword specifies the data to be encoded and the encoding method used. The first eight bytes of the area pointed to by the ENTITY operand are used by the data encryption standard (DES) encoding routine. If ENTITY is not specified, the user ID from the current ACEE is used instead. If TYPE=ENCRYPT is specified, no work area is returned.

, ENCRYPT=(data address,DES)
, ENCRYPT=(data address,HASH)
, ENCRYPT=(data address,INST)
specifies the data to be authenticated, and a method of authentication. The address points to a 1-byte length field followed by 1 to 255 bytes of clear-text data to be used as the user-authentication key. The second subparameter specifies the authentication method: the RACF data encryption standard algorithm, the RACF hashing algorithm, or whatever scheme the installation uses (INST value). Upon return to the macro issuer, the first subparameter contains the address of an area that contains a 1-byte length followed by the encoded version of the data. Neither the address itself nor the length is changed.

Note: When the DES algorithm is used, RACF actually encrypts the data pointed to by the ENTITY profile or by the user ID, using the data as the encryption key. Data is one-way encrypted, that is, no facility is provided to recover the data in readable form. If HASH is specified, the RACF hashing algorithm is used and data is masked instead of encrypted.

, ENTITY=resource name address
specifies the address of an area containing the resource name. The resource name is a 44-byte DASD data set name for CLASS='DATASET', an 8-byte area containing the user ID for CLASS='USER', an 8-byte area containing the group name for CLASS='GROUP', or a 17-byte area for CLASS='CONNECT'. The length of all other resource names is determined from the class descriptor table. The name must be left-justified in the field and padded with blanks. For CLASS='USER', the user ID from the current ACEE or the ACEE specified for ACEE= is used if ENTITY= is not specified.

, ACEE=acee address
specifies an alternate ACEE for RACF to use rather than the current ACEE. For example, if the ENTITY parameter has not been specified, RACF refers to the ACEE during extract processing of user data. If you want to use the ACEE parameter, you must specify RELEASE=1.8 or later.

, VOLSER=vol-address (valid only with ,CLASS='DATASET')
specifies the volume serial as follows:

- For MVS/VSAM DASD data sets and tape data sets, specifies the volume serial number of the catalog controlling the data set.
- For non-VSAM DASD data sets and tape data sets, specifies the volume serial number of the volume on which the data set resides.

The field pointed to by the vol-address variable contains the volume serial number. If necessary, you must pad it to the right with blanks so it contain six characters.

If you specify VOLSER, you must specify RELEASE=1.8 or later.
,GENERIC=ASIS|YES
When CLASS is DATASET, specifies whether RACF is to treat the entity name as a generic profile name.

- If GENERIC=YES is specified, the resource name is considered a generic profile name, even if it does not contain a generic character: an asterisk (*) or a percent sign (%). If you specify GENERIC=YES, the resource name in the macro will match only a generic resource name in the RACF database. It will not match a discrete name.
- If GENERIC=ASIS is specified, the resource name is considered a generic only if it contains a generic character: an asterisk (*) or a percent sign (%).

If you specify GENERIC, you must specify RELEASE=1.8 or later.

,FLDACC=NO|YES
specifies whether field-level access checking should be performed. If you specify FLDACC=YES, the RACF database manager checks to see that the user running your program has the authority to extract or modify the fields that are specified in the RACXTRT macro.

Note:
1. For field-level access checking to occur, you must specify RELEASE=1.8 or later when you code the macro. In addition, before the program executes, the security administrator must activate and RACLIST the FIELD class using the SETROPTS command. If you code FLDACC=YES and the FIELD class is not active and RACLISTed, the request is failed with a return code 8, reason code 4.
2. In addition, the security administrator must issue the RDEFINE and PERMIT commands to designate those users who have the authority to access the fields designated in the RACXTRT macro.
3. If you specify FLDACC=NO or omit the parameter, the manager ignores field-level access checking.

,CLASS='class name'
,CLASS=class name addr
specifies the class the entity is in. The class name can be USER, GROUP, CONNECT, DATASET, or any general-resource class defined in the class descriptor table. If you specify CLASS, you must specify RELEASE=1.8 or later.

,SEGMENT='segment name'
,SEGMENT=segment name address
specifies the RACF profile segment that RACF is to update or from which it is to extract data. If you specify SEGMENT, you must also specify the CLASS and FIELDS keywords, and RELEASE=1.8 or a later release number. If you allow the SEGMENT parameter to default, RACF assumes that you want to extract information from the base segment.

,SEGDATA=segment data addr
specifies the address of a list of data items to be placed in the fields named by the FIELDS= parameter. You use the SEGDATA parameter when you specify TYPE=REPLACE. If you specify SEGDATA, you must also specify CLASS, FIELDS, and RELEASE=1.8 or a later release number. The stored data is paired in the following format:

- A 4-byte length field that contains the length of the data field that follows
- A data field of variable length.
Each length field is followed immediately by a data field until you reach the end of the replacement data. The count field, which is pointed to by the first field in the FIELDS parameter, contains the total number of length-data pairs.

Parameters for RELEASE=1.6 through 1.8.1

The RELEASE values for which a specific parameter is valid are marked with an 'X'. RELEASE=1.6 is the default.

Table 23. RACXTRT parameters for RELEASE=1.6 through 1.8.1

<table>
<thead>
<tr>
<th>Parameter</th>
<th>RELEASE=1.6</th>
<th>RELEASE=1.7</th>
<th>RELEASE=1.8 or 1.8.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACEE=</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLASS=</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DERIVE=YES</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ENCRYPT=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ENTITY=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>EXTRACT=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>EXTRACTN=</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>FLDACC=</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FIELDS=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>GENERIC=</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>RELEASE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>REPLACE=</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>SEGDATA=</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SEGMENT=</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SUBPOOL=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>TYPE=</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VOLSER=</td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

Return codes and reason codes

If the return codes and reason codes you are receiving are not discussed in this macro, refer to the return codes and reason codes described in RACROUTE REQUEST=EXTRACT “Return codes and reason codes” on page 119.

When control is returned, register 15 contains one of the following return codes, and register 0 can contain a reason code.

**Hex Code**

<table>
<thead>
<tr>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
</tr>
</tbody>
</table>

**Reason Code**

**For Derive Requests**

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Some of the values are derived from the USER profile, and some might be derived from the GROUP profile.</td>
</tr>
<tr>
<td>4</td>
<td>High-level qualifier returned as RESOWNER. It matched a valid USER.</td>
</tr>
<tr>
<td>8</td>
<td>DFP data returned from an EXTRACT request from USER profile was actually from the user's default connect group.</td>
</tr>
</tbody>
</table>
0C  High-level qualifier returned as RESOWNER. It matched a valid GROUP.
24  RESOWNER field matched a valid USER.
28  RESOWNER field matched a valid GROUP.
2C  At least one, but not all, of the fields requested failed to be retrieved because of field level access checking.
04  An ESTAE environment was not able to be established, or if register 0 contains a reason code of 1, neither EXTRACT nor ENCRYPT was specified for TYPE=.
08  For TYPE=EXTRACT, TYPE=EXTRACTN, or TYPE=REPLACE the profile cannot be found. The hexadecimal reason codes are:

**Reason code**

**For derive requests**

0  No profile found.
4  Field-level access checking failed. The FIELD class might not be active and RACLISTed.
8  Segment not found.
14  Neither the RESOWNER field nor the high-level qualifier matched a valid USER or GROUP.

C  RACF is inactive.
10  The extract operation failed. Register 0 contains the RACF-manager return code that represents the cause of termination. This return code is not used for the encoding function. The manager return code and reason codes are returned in the low-order and high-order halfwords of register 0.
14  For TYPE=ENCRYPT or TYPE=EXTRACT of user-class data, ENTITY was specified and no ACEE exists, or the ACEE was not for a defined user.

**Reason code**

**For derive requests**

0  No ACEE exists.
4  ACEERACF bit is off.

18  A parameter-list error was encountered. The hexadecimal reason codes are:

**Reason code**

**For derive requests**

4  For TYPE=REPLACE request, FIELDS= was not specified.
8  Type specified is not valid
C  Number of fields not valid.
10  Class name specified is not valid.
14  Version in parameter list is not valid.
18  Subpool specified is not valid.
1C  Parameter length not valid.
20  For TYPE=REPLACE request, SEGDATA= was not specified.
24  Entity name specified is not valid.
RACXTRT

2C For TYPE=ENCRYPT request, no user-authentication key was specified.

30 Encoding method is not valid.

34 ENTITY= was not specified with TYPE=REPLACE, TYPE=EXTRACTN, or TYPE=EXTRACT with class other than USER.

38 Multiple profiles; no volume specified.

3C Profile found wrong volume serial number specified.

48 Entity-name length with the ENTITY keyword not valid:
- The specified length is one of the following:
  - Greater than 44 if CLASS=DATASET
  - Greater than 8 if CLASS=USER or GROUP
  - Greater than 17 if CLASS=CONNECT
  - Greater than the maximum for the specified class as defined in the class descriptor table.
- For TYPE=ENCRYPT request, the specified length is not zero or eight.

50 The entity name contains a blank at the beginning or in the middle of the name.

64 Indicates that the CHECK subparameter of the RELEASE keyword was specified on the execute form of the RACXTRT macro; however, the list form of the macro does not have the proper RELEASE parameter. It also indicates that the TYPE parameters specified on the list and execute forms might not be the same TYPE. Macro processing terminates.

RACXTRT (list form)

The list form of the RACXTRT macro is written as follows:

```
name
name: Symbol. Begin name in column 1.

b
One or more blanks must precede RACXTRT.

RACXTRT
b
One or more blanks must follow RACXTRT.

TYPE=EXTRACT
TYPE=EXTRACTN
TYPE=REPLACE
TYPE=ENCRYPT

,ENTITY=resource name
addr
resource name addr: A-type address

,ACEE=acee-address
acee: A-type address

,VOLSER=volser-address
vol address: A-type address

,GENERIC=ASIS
,GENERIC=YES
Default: ASIS
```
If TYPE=EXTRACT or EXTRACTN is specified:

- **SUBPOOL=** subpool number
  - Decimal digit 0–255
  - Default: SUBPOOL=229

- **DERIVE=YES**
  - See explanation of keyword.
  - Default: Normal processing

- **CLASS=’class name’**
  - class name: 1–8 character class name
  - Default: CLASS=’USER’

- **SEGMENT=’segment name’**
  - segment name: 1–8 character name

- **FIELDS=** field addr
  - field addr: A-type address

If TYPE=REPLACE is specified:

- **CLASS=’class name’, class name addr**
  - class name addr: Rx-type address or Register (2) – (12)
  - Default: CLASS=’USER’

- **SEGMENT=’segment name’, segment name addr**
  - segment name addr: A-type address

- **FIELDS=** field addr
  - field addr: A-type address

- **SEGDATA=** segment data addr
  - segment data addr: A-type address

If TYPE=ENCRYPT is specified:

- **ENCRYPT=(data address,DES)**
- **ENCRYPT=(data address,HASH)**
- **ENCRYPT=(data address,INST)**

The parameters are explained under the standard form of the RACXTRT macro with the following exception:

- **MF=L**
  - specifies the list form of the RACXTRT macro.
The execute form of the RACXTRT macro is written as follows:

```
name
```

name: Symbol. Begin name in column 1.

b

One or more blanks must precede RACXTRT.

RACXTRT

b

One or more blanks must follow RACXTRT.

**TYPE=EXTRACT**

**TYPE=ENCRYPT**

```
,ENTITY=resource name
   addr

,RELEASE=number

,RELEASE=(,CHECK)

,RELEASE=(number,CHECK)

,ACEE=acee-address

,VOLSER=volser-address

,GENERIC=ASIS

,GENERIC=YES

,FLDACC=YES

,FLDACC=NO

,MF=(E,ctrl addr)
```

ctrl addr: Rx-type address, register (1) or register (2) – (12)

If **TYPE=EXTRACT or EXTRACTN** is specified:

```
,SUBPOOL=subpool
   number

,DERIVE=YES
```

See explanation of keyword.

```
,CLASS=class name addr
```

class name addr: Rx-type address or register (2) – (12)

```
,SEGMENT=segment
   name addr
```

segment name addr: Rx-type address or register (2) – (12)

```
,FIELDS=field addr
```

field addr: Rx-type address or register (2) – (12)

If **TYPE=REPLACE** is specified:

```
,CLASS=class name addr
```

class name addr: Rx-type address or Register (2) – (12)

```
,SEGMENT=segment
   name addr
```

segment name addr: Rx-type address or register (2) – (12)

```
,FIELDS=field addr
```

field addr: Rx-type address or register (2) – (12)
If TYPE=ENCRYPT is specified:

- ENCRYPT=(data address,DES)
- ENCRYPT=(data address,HASH)
- ENCRYPT=(data address,INST)

The parameters are explained under the standard form of the RACXTRT macro with the following exceptions:

- RELEASE=number
- RELEASE=(,CHECK)
- RELEASE=(number,CHECK)

specifies the RACF release level of the parameter list to be generated by this macro.

Certain parameters can be specified only with particular releases. If you specify a parameter with an incompatible release level, the parameter is not accepted by the macro processing. An error message is issued at assembly time. For the parameters that are valid for RELEASE=1.6 and later, see Table 23 on page 340.

When you specify the RELEASE keyword, checking is done at assembly time. Compatibility between the list and execute forms of the RACXTRT macro is validated at execution time if you specify the CHECK subparameter on the execute form of the macro.

The size of the list form expansion must be large enough to accommodate all parameters defined by the RELEASE keyword on the execute form of the macro. Otherwise, when CHECK processing is requested, the execute form of the macro is not done, and a return code of X’64’ is returned.

The default is RELEASE=1.6.

- MF=(E,ctrl addr)

specifies the execute form of the RACXTRT macro, using a remote, control-program parameter list.
RACXTRT
Appendix B. RACF database templates

This topic includes information about the following templates of the RACF database:
- GROUP
- USER
- CONNECT
- DATA SET
- GENERAL
- RESERVED

Important

Do not modify the RACF database templates (CSECT IRRTEMP2). Such modification is not supported by IBM and might result in damage to your RACF database or other unpredictable results.

Segment fields:
1. The first field in a segment of a template cannot be retrieved or updated. This field has a Field ID of 001 and is usually described in the Field Being Described column as ‘Start of segment fields’.
2. The TME segment fields are intended to be updated by Tivoli® applications, which manage updates, permissions, and cross references among the fields. The TME fields should only be directly updated on an exception basis. See z/OS Security Server RACF Command Language Reference for formats of the field data as enforced by the RACF commands. Use caution when directly updating TME fields, as the updates might be overridden by subsequent actions of Tivoli applications.

Format of field definitions

The RACF database templates contain a definition for each field in the profile.

Each field definition contains information about the field in the following format:

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field Length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
</table>

Field Name: Character data.
Field ID: Reference number.
Flag 1 field: The bits have the following meanings when they are turned on:
- Bit 0: The field is a member of a repeat group.
- Bit 1: The definition describes a combination field.
- Bit 2: The field is a flag byte.
- Bit 3: The field contains the count of members in the repeat group following this field.
- Bit 4: The definition describes a combination field continued in next entry.
- Bit 5: The field (for example, PASSWORD) is masked.
Database templates

Bit 6: The field is sorted in ascending order.
Bit 7: The field is a statistical field. A value is always stored for this field, even when it is equal to the defined null value for the field.

Flag 2
The bits have the following meanings when they are turned on:

Bit 0: Changes to this field affect security and cause ACEEs to be purged from VLF.
Bit 1: The field is padded on the left with binary zeros when values shorter than the field length are retrieved.
Bit 2: This field represents a 3-byte date field.
Bit 3: This field is an Application Identity Mapping alias name.
Bit 4: This field is not to be unloaded by the Database Unload utility (IRRDBU00).
Bit 5: The alias name in this field is EBCDIC.
Bits 6–7: Reserved for IBM’s use.

Field Length
Field length on return from ICHEINTY or RACROUTE REQUEST=EXTRACT (0 is variable length).

Default Value
Field default. If the field is not present in the profile, this byte is propagated throughout the returned field as the default value.

Type
Data type of each field. In this column, character is represented as 'Char', integer is represented as 'Int', and binary is represented as 'Bin'. 'Date' and 'Time' are also possible data types.

The type of a combination field that represents a single field is the same as that single field. There is no "type" associated with a combination field which represents multiple fields.

Repeat groups on the RACF database
A repeat group consists of one or more sequential fields within a profile that are able to be repeated within that profile. A field that belongs to a repeat group is only defined once in the template, but can be repeated as many times as necessary within the actual profile. A count field precedes the repeat group in the profile indicating how many of these groups follow.

Field length
If a field in a profile has a fixed length, a value (less than 255) in the field definition within the template specifies its actual length. If a field in a profile has a variable length, the value in the field definition is 0. In both cases, the actual field length is contained in the physical data mapped by the field definition.

Data field types
RACF stores information in the RACF database in many different formats. This section identifies the major data types that RACF stores. Exceptions and additional detail can be found in the description of each specific field within the templates.

Date fields
The format of the 3-byte date fields is yydddf, which represents a packed decimal number in which y represents year, d represents day, and F represents the sign. Examples of RACF date values are X’98111C’ and X’94099D’.
The format of the 4-byte date fields should be yyyymmdd, which represents a packed decimal number in which y represents year, m represents month, and d represents day. Examples of RACF date values are X'19980421' and X'19940409'.

RACF might use any of the following values for null dates: X'FFFFFF', X'00000D', X'00000C', and X'000000' for 3-byte addresses, and X'FFFFFFFF', X'0000000D', X'0000000C', and X'00000000' for 4-byte addresses. However, you should always set null dates to either X'00000F' for 3-byte addresses and X'0000000F' for 4-byte addresses.

**Time fields**
The format for the 4-byte time fields are hhmmssstc where h represents hours, m represents minutes, s represents seconds, t represents tenths of seconds, and c represents hundredths of seconds. There is no sign byte. For information on the 8-byte version, see the TIME macro as documented in [z/OS MVS Programming: Assembler Services Reference IAR-XCT]

**Integer fields**
Integers are stored as unsigned binary values. These values can be 1, 2, or 4 bytes in length.

**Character fields**
Character fields are padded with blanks to the right.

### Combination fields on the RACF database
The database templates also contain definitions called *combination fields*.

Combination fields do not describe a field of a profile. They contain the field numbers that identify the respective field definitions. You can use a combination as an alias to access multiple fields with one ICHEACTN or RACROUTE REQUEST=EXTRACT macro. For more information, see "Example 2: Adding a user ID to a data set access list", in Appendix A of [z/OS Security Server RACF Macros and Interfaces]

In addition, you can use the combination field to provide aliases for individual fields.

The format of a combination field definition is different from a non-combination definition. Its format is as follows:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Character data.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field ID</td>
<td>Reference number.</td>
</tr>
<tr>
<td>Flag 1</td>
<td>The hex representation of the flag bits for this field. For combination fields, bit 1 is on. For a continuation of combination fields, bit 4 is also on.</td>
</tr>
<tr>
<td>Flag 2</td>
<td>The hex representation of the flag bits for this field. For combination fields, all bits are off.</td>
</tr>
<tr>
<td>Combination IDs</td>
<td>If nonzero, combination IDs represent the position of a non-combination field within the template segment. There can be up to 5 field IDs representing the template fields that comprise this combination field.</td>
</tr>
<tr>
<td>Type</td>
<td>Data type of each field. In this column, character is represented as 'Char', integer is represented as 'Int', and binary is represented as 'Bin'. 'Date' and 'Time' are also possible data types.</td>
</tr>
<tr>
<td>Comments</td>
<td>Comment field.</td>
</tr>
</tbody>
</table>
Determining space requirements for the profiles

The formula for calculating the space required for each segment (Base RACF information, TSO, DFP, and so on) of each profile in the RACF database is as follows:

\[ P = 20 + L + F1 + F4 + R \]

Where:

- \( P \) = The number of bytes required for a profile segment
- \( L \) = The number of bytes in the profile name
- \( F1 \) = The sum of the lengths of all fields that contain data and have a length of 1 to 127 bytes, plus 2 bytes for every field counted.
  
  For example, if a segment contains 3 non-null fields of length 8, \( F1 = (3 \times 8) + (3 \times 2) = 24 + 6 = 30 \).

- \( F4 \) = The sum of the lengths of all fields that contain data and have a length of 128 to \( 2^{31} \) bytes, plus 5 bytes for every field counted.
  
  For example, if a segment contains a non-null field 150 bytes long and a non-null field 255 bytes long, \( F4 = 150 + 255 + (2 \times 5) = 150 + 255 + 10 = 415 \).

- \( R \) = The sum of the lengths of all repeat groups. If a repeat group has no occurrences, then it has a length of 0 bytes. If a repeat group has 1 or more occurrences, then the length of each repeat group is calculated as follows:
  
  \[ 9 + N + G1 + G4 \]

  - \( N \) = The number of occurrences of the group
  - \( G1 \) = The sum of the lengths of all fields in the group, which have a length of 1 to 127 bytes, plus 1 byte for every field counted. If a field has a length of zero, it will still take up 1 byte in the profile.
  - \( G4 \) = The sum of the lengths of all fields in the group, which have a length of 128 to \( 2^{31} \) bytes, plus 4 bytes for every field counted.

  For example, consider a group with two occurrences. Each occurrence contains an 8-byte field and a variable length field. In the first occurrence, the variable length field is 30 bytes and in second occurrence, it is 200 bytes. The length of the group is:

  \[ 9 + 2 + G1 + G4 \]

  \( G1 \) is \((8 + 1) + (30 + 1)\) from the first occurrence and \((8 + 1)\) from the second, for a total of 49 bytes. \( G4 \) is \((200 + 4)\) from the second occurrence, or 204 bytes. So, the length of the group is \( 9 + 2 + 49 + 204 \), or 264 bytes.

Note: For each repeat group (except CGGRPCT in the USER profile), the amount of data can not exceed 65,535 bytes to ensure proper processing by programs retrieving the data using ICHEINTY with DATAMAP=OLD. To calculate the amount of data to determine whether it will fit within this limit, examine the template definitions for the repeat group and the data for that repeat group contained within the profile. For each fixed length field in each occurrence of the repeat group add the length of the field as shown in its template definition. For each variable length field in each occurrence of the repeat group add the length of the data in the field plus one. When you are done, the total cannot exceed 65,535.
For example, this would translate into a maximum of 8191 group connections per user, based on the CONGRPCT repeat group in the USER template. This group contains one 8-byte field, making the calculation of the limit a simple one of dividing 65 535 by 8 and dropping any remainder.

As another example, this would translate into a maximum of 5957 users connected to a group, based on the ACLCNT repeat group in the GROUP template. This group contains one 8-byte field (USERID), one 1-byte field (USERACS), and one 2-byte field (ACSCNT). This gives a total length of eleven for the fixed-length fields in each occurrence. Dividing 65 535 by 11 and dropping the remainder gives the limit of 5957.

When calculating F1 and F4, remember that statistical fields (Flag1/bit 7 on, in the template definition) are always stored in a profile segment, even when the field contains a null value. For example, LJTIME always adds 3 bytes to the length of a USER profile Base segment, regardless of whether it contains a zero value or some other value. Other fields only exist in the segment if a specific value has been added for that field.

Note: The RACF database space required for a segment is a multiple of the 256-byte slots required to contain the segment. For example, if a USER profile Base segment contains 188 bytes of data, it still requires 256 bytes of space in the RACF database.

Determining space requirements for alias index entries

An exact formula for calculating the space required for alias entries cannot be derived due to index block compression, and the mechanics of the higher level index blocks. The following is an approximate formula for space taken up in the alias index block sequence set (level 1) by an index entry:

\[
AIE = 16 + (N \times (2 + BPL)) + AKL
\]

Where:

- **AIE** = Alias index entry length
- **N** = The number of base profile names. Most aliases are only allowed 1 base profile association.
- **BPL** = Base profile name length. The base profile is named in the alias index entry. Because the base profile is a user or group profile, valid BPL values are between 1 and 8.
- **AKL** = Alias index entry, key length. The first 3 bytes are template number, segment number, and field number. Additional bytes of the alias index key are taken up by the alias value. These lengths vary according to each alias field.

Group template for the RACF database

The group template describes the fields of group profiles in the RACF database.

<table>
<thead>
<tr>
<th>FIELD</th>
<th>ACSCNT</th>
<th>FLDCNT</th>
<th>FLDNAME</th>
<th>INITCNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLDVALUE</td>
<td>FLDFLAG</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

End of NOT programming interface information
**Group template**

**Note:**

1. Application developers should not depend on being able to use RACROUTE REQUEST=EXTRACT for the BASE segment fields on any security product other than RACF. These products are expected to support only such segments as DFP and TSO.

2. The TME segment fields are intended to be updated by the Tivoli applications, which manage updates, permissions, and cross-references among the fields. The TME fields should only be directly updated on an exception basis. See [z/OS Security Server RACF Command Language Reference](#) for formats of the field data as enforced by the RACF commands. Use caution when directly updating TME fields, as the updates might be overridden by subsequent actions of Tivoli applications.

The contents of the group template are as follows:

<table>
<thead>
<tr>
<th>Template Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>GROUP 001</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>ENTYPE 002</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>01</td>
<td>Int</td>
</tr>
<tr>
<td>VERSION 003</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>01</td>
<td>Int</td>
</tr>
<tr>
<td>SUPGROUP 004</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>FF</td>
<td>Char</td>
</tr>
<tr>
<td>AUTHDATE 005</td>
<td>00</td>
<td>20</td>
<td>00</td>
<td>00000003</td>
<td>FF</td>
<td>Date</td>
</tr>
<tr>
<td>AUTHOR 006</td>
<td>00</td>
<td>00</td>
<td>80</td>
<td>00000008</td>
<td>FF</td>
<td>Char</td>
</tr>
<tr>
<td>INTCNT 007</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00000002</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>UACC 008</td>
<td>20</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>NOTRMUAC 009</td>
<td>20</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>INSTDATA 010</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>MODELNAME 011</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>FLDcnt 012</td>
<td>10</td>
<td>00</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>FLDNAME 013</td>
<td>80</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>FLDValue 014</td>
<td>80</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>FLDFLAG 015</td>
<td>A0</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>SUBGRPCT 016</td>
<td>10</td>
<td>00</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>SUBGRPNM 017</td>
<td>80</td>
<td>80</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>ACLCNT 018</td>
<td>10</td>
<td>00</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>USERID 019</td>
<td>80</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
</tbody>
</table>

*Template Field being described*

- **GROUP 001**: The number (1) corresponding to group profiles.
- **ENTYPE 002**: The version field from the profile. Always X'01'.
- **VERSION 003**: The superior group to this group.
- **SUPGROUP 004**: The date the group was created.
- **AUTHOR 006**: The owner (user ID or group name) of the group.
- **AUTHDATE 005**: Reserved for IBM's use.
- **NOTRMUAC 009**: The universal group authority. (The authority of a user to the group if the user is not connected to the group.)
- **INTCNT 007**: Reserved for IBM's use.
- **UACC 008**: Installation data.
- **MODELNAM 011**: Data set model profile name. The profile name begins with the second qualifier; the high-level qualifier is not stored.
- **FLDCNT 012**: Reserved for IBM's use.
- **FLDNAME 013**: Reserved for IBM's use.
- **FLDVALUE 014**: Reserved for IBM's use.
- **FLDFLAG 015**: Reserved for IBM's use.
- **SUBGRPCT 016**: The number of subgroups of the group.
- **SUBGRPNM 017**: A list of the subgroup names.
- **ACLINT 018**: The number of users connected to the group.
- **USERID 019**: The user ID of each user connected to the group.
## GROUP template

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>USERACS</td>
<td>020</td>
<td>A0</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>ACSCNT</td>
<td>021</td>
<td>80</td>
<td>00</td>
<td>00000002</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>USRCNT</td>
<td>022</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>USRNMM</td>
<td>023</td>
<td>80</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>USRDATA</td>
<td>024</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>USRFLG</td>
<td>025</td>
<td>A0</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>UNVFLG</td>
<td>026</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
</tbody>
</table>

### Field being described

The group authority of each user connected to the group.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>JOIN authority</td>
</tr>
<tr>
<td>1</td>
<td>CONNECT authority</td>
</tr>
<tr>
<td>2</td>
<td>CREATE authority</td>
</tr>
<tr>
<td>3</td>
<td>USE authority</td>
</tr>
<tr>
<td>4–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

**Note 1:** Intended usage for these fields is to allow the installation to store additional data in this profile. USRNM should have a field name to use as a key to identify each unique occurrence of a row in the repeat group. USRDATA and USRFLG hold the data associated with that name. For more information, see "Example 5: Updating the installation fields", in Appendix A of *z/OS Security Server RACF Macros and Interfaces*.
Group template

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSDATA</td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Start of segment fields for custom fields

Note: Intended usage for these fields is dictated by your installation. See z/OS Security Server RACF Security Administrator’s Guide for more information on custom fields.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSCNT</td>
<td>002</td>
<td>10</td>
<td>00</td>
<td>4</td>
<td>0</td>
<td>Integer</td>
</tr>
<tr>
<td>CSTYPE</td>
<td>003</td>
<td>80</td>
<td>00</td>
<td>1</td>
<td>0</td>
<td>Bin</td>
</tr>
</tbody>
</table>

Count of custom fields

Custom field type:
- 01 - character
- 02 - numeric
- 03 - flag
- 04 - hex

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Combination field IDs</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSKEY</td>
<td>004</td>
<td>80</td>
<td>00</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>CSVALUE</td>
<td>005</td>
<td>80</td>
<td>00</td>
<td>0</td>
<td>Char</td>
</tr>
</tbody>
</table>

Custom field keyword; maximum length = 8

Custom field value

User template for the RACF database

The user template describes the fields of the user profiles in a RACF database.

---

**NOT programming interface information**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CATEGORY</td>
<td>FIELD</td>
</tr>
<tr>
<td>CONGRPCT</td>
<td>FLDCNT</td>
</tr>
<tr>
<td>CONGRPNM</td>
<td>FLDFLAG</td>
</tr>
<tr>
<td>CURKEY</td>
<td>FLDNAME</td>
</tr>
<tr>
<td>CURKEYV</td>
<td>FLDVALUE</td>
</tr>
<tr>
<td>ENCTYPE</td>
<td>MAGSTRIP</td>
</tr>
<tr>
<td>ENCTYPE</td>
<td>NUMCTGY</td>
</tr>
<tr>
<td>ENCTYPE</td>
<td>PPHENV</td>
</tr>
<tr>
<td>ENCTYPE</td>
<td>PPHENV</td>
</tr>
<tr>
<td>ENCTYPE</td>
<td>PPHENV</td>
</tr>
<tr>
<td>ENCTYPE</td>
<td>PHRCNT</td>
</tr>
<tr>
<td>ENCTYPE</td>
<td>PWDENV</td>
</tr>
<tr>
<td>ENCTYPE</td>
<td>PWDENV</td>
</tr>
<tr>
<td>ENCTYPE</td>
<td>PWDENV</td>
</tr>
<tr>
<td>ENCTYPE</td>
<td>PWDENV</td>
</tr>
<tr>
<td>ENCTYPE</td>
<td>PWDENV</td>
</tr>
<tr>
<td>ENCTYPE</td>
<td>SALT</td>
</tr>
</tbody>
</table>

---

**End of NOT programming interface information**

**Note:** Application developers should not depend on being able to use RACROUTE REQUEST=EXTRACT for the BASE segment fields on any security product other than RACF. These products are expected to support only such segments as DFP and TSO.

The contents of the user template (base segment) are as follows:

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>USER</td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>0</td>
<td>Int</td>
</tr>
<tr>
<td>ENTYPE</td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>02</td>
<td>Int</td>
</tr>
<tr>
<td>VERSION</td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>01</td>
<td>Int</td>
</tr>
</tbody>
</table>

The number (2) corresponding to user profiles.

The version field from the profile. Always X’01’.
### User template

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
<th>Field being described</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTHDATE</td>
<td>004</td>
<td>00</td>
<td>20</td>
<td>00000003</td>
<td>FF</td>
<td>Date</td>
<td>The date the user was defined to RACF.</td>
</tr>
<tr>
<td>AUTHOR</td>
<td>005</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>FF</td>
<td>Char</td>
<td>The owner (user ID or group name) of the user profile.</td>
</tr>
<tr>
<td>FLAG1</td>
<td>006</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
<td>Identifies the user as having (bit 0 is on) or not having the ADSP attribute.</td>
</tr>
<tr>
<td>FLAG2</td>
<td>007</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
<td>Identifies the user as having (bit 0 is on) or not having the SPECIAL attribute.</td>
</tr>
<tr>
<td>FLAG3</td>
<td>008</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
<td>Identifies the user as having (bit 0 is on) or not having the OPERATIONS attribute.</td>
</tr>
<tr>
<td>FLAG4</td>
<td>009</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
<td>Identifies the user as having (bit 0 is on) or not having the REVOKE attribute.</td>
</tr>
<tr>
<td>FLAG5</td>
<td>010</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
<td>Identifies the user as having (bit 0 is on) or not having the GRPACC attribute.</td>
</tr>
<tr>
<td>PASSINT</td>
<td>011</td>
<td>00</td>
<td>80</td>
<td>00000001</td>
<td>FF</td>
<td>Int</td>
<td>The interval in days (represented by a number between 1 and 254) that the user’s password is in effect. If it is X’FF’, the user’s password never expires. See the description of the SETR PASSWORD(INTERVAL,...)) processing instructions in z/OS Security Server RACF Command Language Reference for more details.</td>
</tr>
<tr>
<td>PASSWORD</td>
<td>012</td>
<td>04</td>
<td>80</td>
<td>00000008</td>
<td>FF</td>
<td>Char</td>
<td>The password associated with the user. For masking, the masked password is stored. For DES, the encrypted user ID is stored. If the installation provides its own password authentication, data returned by the ICHDEX01 exit is stored.</td>
</tr>
<tr>
<td>PASSDATE</td>
<td>013</td>
<td>00</td>
<td>20</td>
<td>00000003</td>
<td>FF</td>
<td>Date</td>
<td>The date the password was last changed.</td>
</tr>
<tr>
<td>PGMRNAME</td>
<td>014</td>
<td>00</td>
<td>00</td>
<td>00000020</td>
<td>FF</td>
<td>Char</td>
<td>The name of the user.</td>
</tr>
<tr>
<td>DFLTGRP</td>
<td>015</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>FF</td>
<td>Char</td>
<td>The default group associated with the user. A value of X’FF’ indicates that no group was specified.</td>
</tr>
<tr>
<td>LJTIME</td>
<td>016</td>
<td>01</td>
<td>00</td>
<td>00000004</td>
<td>FF</td>
<td>Time</td>
<td>The last recorded time that the user entered the system by using RACROUTE REQUEST=VERIFY.</td>
</tr>
<tr>
<td>LJDATE</td>
<td>017</td>
<td>01</td>
<td>20</td>
<td>00000003</td>
<td>FF</td>
<td>Date</td>
<td>The last recorded date that the user entered the system by using RACROUTE REQUEST=VERIFY.</td>
</tr>
<tr>
<td>INSTDATA</td>
<td>018</td>
<td>00</td>
<td>80</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Installation data.</td>
</tr>
<tr>
<td>UAUDIT</td>
<td>019</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
<td>Identifies whether all RACROUTE REQUEST=AUTH, RACROUTE REQUEST=DEFINE, (and, if the caller requests logging, RACROUTE REQUEST=FASTAUTH) macros issued for the user and all RACF commands (except SEARCH, LISTDS, LISTGRP, LISTUSER, and RLIST) issued by the user is logged. If bit 0 is on, they are logged. If bit 0 is off, logging might still occur for other reasons, as identified in z/OS Security Server RACF Auditor’s Guide.</td>
</tr>
<tr>
<td>FLAG6</td>
<td>020</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
<td>Identifies the user as having (bit 0 is on) or not having the AUDITOR attribute.</td>
</tr>
</tbody>
</table>
### User template

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLAG7</td>
<td>021</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>FLAG8</td>
<td>022</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>MAGSTRIP</td>
<td>023</td>
<td>04</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>PWDDGEN</td>
<td>024</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>PWDCNT</td>
<td>025</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>OLDPWDNM</td>
<td>026</td>
<td>80</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>OLDPWD</td>
<td>027</td>
<td>84</td>
<td>00</td>
<td>00000008</td>
<td>FF</td>
<td>Char</td>
</tr>
<tr>
<td>REVOKECT</td>
<td>028</td>
<td>01</td>
<td>80</td>
<td>00000001</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>MODELNAME</td>
<td>029</td>
<td>00</td>
<td>80</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>SecLevel</td>
<td>030</td>
<td>00</td>
<td>80</td>
<td>00000001</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>NUMCTGY</td>
<td>031</td>
<td>10</td>
<td>80</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>CATEGORY</td>
<td>032</td>
<td>80</td>
<td>80</td>
<td>00000002</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>REVOKEEDT</td>
<td>033</td>
<td>00</td>
<td>20</td>
<td>00000000</td>
<td>00</td>
<td>Date</td>
</tr>
<tr>
<td>RESUMEDT</td>
<td>034</td>
<td>00</td>
<td>20</td>
<td>00000000</td>
<td>00</td>
<td>Date</td>
</tr>
<tr>
<td>LOGDAYS</td>
<td>035</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>LOGTIME</td>
<td>036</td>
<td>00</td>
<td>80</td>
<td>00000000</td>
<td>00</td>
<td>Time</td>
</tr>
<tr>
<td>FLDCNT</td>
<td>037</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>FLDNAME</td>
<td>038</td>
<td>80</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>FLDVVALUE</td>
<td>039</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>FLDFLAG</td>
<td>040</td>
<td>A0</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>CLCNT</td>
<td>041</td>
<td>10</td>
<td>80</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
</tbody>
</table>

### Field being described

- **FLAG7**: If bit 0 is on, and FLAG8 has bit 0 on, an operator identification card (OID card) is needed to enter the system.
- **FLAG8**: If bit 1 is on, this is a protected user ID, which cannot enter the system by any means requiring a password or OID card.
- **MAGSTRIP**: The operator identification associated with the user from the masked or encrypted OID card data required to authenticate this user, as supplied by a supported 327x (such as 3270 and 3278) OID card reader.
- **PWDDGEN**: Current password generation number.
- **PWDCNT**: Number of old passwords present.
- **OLDPWDNM**: Generation number of previous password.
- **OLDPWD**: Previous password. This is an encrypted password value.
- **REVOKECT**: Count of unsuccessful password attempts. **Note**: You can use ALTER when setting this field, but you cannot use ALTERI.
- **MODELNAME**: Data set model profile name. The profile name begins with the second qualifier; the high-level qualifier is not stored.
- **SECLEVEL**: The number that corresponds to the user's security level. For more information on security levels, see [z/OS Security Server RACF Security Administrator’s Guide](#). **Note**: You can use ALTER when setting this field, but you cannot use ALTERI.
- **NUMCTGY**: Number of security categories.
- **CATEGORY**: A number that corresponds to the security categories to which the user has access.
- **REVOKEEDT**: The date the user is revoked. This field either has length 0, or contains a 3-byte revoke date.
- **RESUMEDT**: The date the user is resumed. This field either has length 0, or contains a 3-byte resume date.
- **LOGDAYS**: The days of the week the user cannot log on (Bit 0 of this field equals Sunday, bit 1 equals Monday, and so on).
- **LOGTIME**: The time of the day the user can log on. If present (length of variable field not equal to 0), it is specified as 6 bytes formatted as two 3-byte packed decimal fields,

\[ \text{0sssCC} \], where \text{ssss} represents the start time (hh:mm) from the \text{ALU\_WHEN\_TIMES\_\(\ldots\)) specification and \text{cccc} represents the end time. For hh:mm, hh represents hours, and mm represents minutes.

- **FLDCNT**: Reserved for IBM’s use.
- **FLDNAME**: Reserved for IBM’s use.
- **FLDVVALUE**: Reserved for IBM’s use.
- **FLDFLAG**: Reserved for IBM’s use.
- **CLCNT**: The number of classes in which the user is allowed to define profiles.
<table>
<thead>
<tr>
<th>Template Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLNAME</td>
<td>042</td>
<td>80</td>
<td>80</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>CONGRPCT</td>
<td>043</td>
<td>10</td>
<td>80</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>CONGRPNM</td>
<td>044</td>
<td>80</td>
<td>80</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>USRCNT</td>
<td>045</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>USRNMT</td>
<td>046</td>
<td>80</td>
<td>80</td>
<td>00000008</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>USRDATA</td>
<td>047</td>
<td>80</td>
<td>80</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>USRFLG</td>
<td>048</td>
<td>A0</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>SECLABEL</td>
<td>049</td>
<td>00</td>
<td>80</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>CGGRPCT</td>
<td>050</td>
<td>10</td>
<td>80</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>CGGRPNM</td>
<td>051</td>
<td>82</td>
<td>80</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>CGAUTHDA</td>
<td>052</td>
<td>80</td>
<td>A0</td>
<td>00000003</td>
<td>FF</td>
<td>Date</td>
</tr>
<tr>
<td>CGAUTHOR</td>
<td>053</td>
<td>80</td>
<td>80</td>
<td>00000008</td>
<td>FF</td>
<td></td>
</tr>
<tr>
<td>CGLTIME</td>
<td>054</td>
<td>81</td>
<td>00</td>
<td>00000004</td>
<td>FF</td>
<td></td>
</tr>
<tr>
<td>CGLDATE</td>
<td>055</td>
<td>81</td>
<td>20</td>
<td>00000003</td>
<td>FF</td>
<td></td>
</tr>
<tr>
<td>CGUACC</td>
<td>056</td>
<td>A0</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>CGINITCT</td>
<td>057</td>
<td>81</td>
<td>00</td>
<td>00000002</td>
<td>FF</td>
<td></td>
</tr>
<tr>
<td>CGFLAG1</td>
<td>058</td>
<td>A0</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>CGFLAG2</td>
<td>059</td>
<td>A0</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>CGFLAG3</td>
<td>060</td>
<td>A0</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>CGFLAG4</td>
<td>061</td>
<td>A0</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>CGFLAG5</td>
<td>062</td>
<td>A0</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>CGNOTUAC</td>
<td>063</td>
<td>A0</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>CGGRPAUD</td>
<td>064</td>
<td>A0</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>CGREVKDT</td>
<td>065</td>
<td>80</td>
<td>20</td>
<td>00000000</td>
<td>00</td>
<td>Date</td>
</tr>
<tr>
<td>CGRESMDT</td>
<td>066</td>
<td>80</td>
<td>20</td>
<td>00000000</td>
<td>00</td>
<td>Date</td>
</tr>
</tbody>
</table>

**Field being described**

- **A class in which the user is allowed to define profiles. (The user has the CLAUTH attribute.)** The user can also define profiles in any other classes with POSIT values matching these classes.
- **The number of groups that the user is connected to.**
- **A group that the user is connected to.**
- **Reserved for installation use.**
  - **Note:** Intended usage: For installation to store additional data in this profile. USRNM should have a field name to use as a key to identify each unique occurrence of a row in the repeat group. USRDATA and USRFLG hold the data associated with that name. For more information, see "Example 5: Updating the installation fields", in Appendix A of *z/OS Security Server RACF Macros and Interfaces*.
- **Security label.**
- **Number of Connect Group entries.** Information from the following CGxxx fields is also available through the logical connect profiles (ICHEINTY with CLASS=CONNECT) in the database. See "Connect template for the RACF database" on page 365 for more details.
- **If bit 0 is on, the user has the ADSP attribute in that group.**
- **If bit 0 is on, the user has the SPECIAL attribute in that group.**
- **If bit 0 is on, the user has the OPERATIONS attribute in that group.**
- **If bit 0 is on, the user has the REVOKE attribute in that group.**
- **If bit 0 is on, the user has the GRPACC attribute in that group.**
- **If bit 0 is on, the user must be specifically authorized (by the PERMIT command) to use a terminal. If off, RACF uses the terminal's UACC.**
- **If bit 0 is on, the user has the GROUP AUDITOR attribute in that group.**
- **The date the user is revoked. This field either has length 0, or contains a 3-byte revoke date.**
- **The date the user is resumed. This field either has length 0, or contains a 3-byte resume date.**
### User template

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length (decimal)</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>TUCNT</td>
<td>067</td>
<td>10</td>
<td>00</td>
<td>00000002</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>TUKEY</td>
<td>068</td>
<td>80</td>
<td>00</td>
<td>00000016</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>TUDATA</td>
<td>069</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Field being described**

- **TUCNT**: Number of user ID associations.
- **TUKEY**: Associated node and user ID.

**Byte Meaning when set**

- **0–7**: The associated node name.
- **8–15**: The associated user ID.

**Associated user ID association data**

- **Byte Meaning when set**
  - **0**: Version number of the TUDATA entry.
  - **1**: Bitstring
    - **0**: Specifies the user as having (bit is on) or not having (bit is off) a peer user ID association.
    - **1**: Specifies the user as being (bit is on) the manager of a managed user ID association.
    - **2**: Specifies the user as being (bit is on) managed by a managed user ID association.
    - **3**: An association request for this user is pending (bit is on) on a remote RRSF node.
    - **4**: An association request for this user is pending (bit is on) on the local RRSF node.
    - **5**: Specifies that password synchronization is in effect (bit is on) for this peer-user ID association.
    - **6**: Specifies that the association request for this user was rejected (bit is on).
    - **7**: Reserved for IBM’s use.

- **Date 2–20**: Reserved for IBM’s use.
- **Time 25–32**: The date the user ID association was defined. (yyyymmd)
<table>
<thead>
<tr>
<th>Template Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
<th>Field being described</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field ID</td>
<td>Flag 1</td>
<td>Flag 2</td>
<td>Field length decimal</td>
<td>Default value</td>
<td>Type</td>
<td>Field being described</td>
<td></td>
</tr>
<tr>
<td>CERTCT</td>
<td>070</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Char</td>
<td>Number of certificate names.</td>
</tr>
<tr>
<td>CERTNAME</td>
<td>071</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Name of certificate. Names correspond to profiles in the DIGTCERT class for the user.</td>
</tr>
<tr>
<td>CERTLABL</td>
<td>072</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Label associated with the certificate.</td>
</tr>
<tr>
<td>CERTSJDN</td>
<td>073</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Subject's distinguished name.</td>
</tr>
<tr>
<td>CERTPU8K</td>
<td>074</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Public key associated with the certificate.</td>
</tr>
<tr>
<td>CERTRSV3</td>
<td>075</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Reserved for IBM's use.</td>
</tr>
<tr>
<td>FLAG9</td>
<td>076</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Char</td>
<td>Restricted Access = BIT0.</td>
</tr>
<tr>
<td>NMAPCT</td>
<td>077</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Char</td>
<td>Number of DIGTNMAP Mapping Profiles that specify this user ID.</td>
</tr>
<tr>
<td>NMAPLABL</td>
<td>078</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Label associated with this mapping.</td>
</tr>
<tr>
<td>NMAPNAME</td>
<td>079</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Name of mapping profile. The names correspond to profiles in the DIGTNMAP class.</td>
</tr>
<tr>
<td>NMAPRSV1</td>
<td>080</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Reserved for IBM's use.</td>
</tr>
<tr>
<td>NMAPRSV2</td>
<td>081</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Reserved for IBM's use.</td>
</tr>
<tr>
<td>NMAPRSV3</td>
<td>082</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Reserved for IBM's use.</td>
</tr>
<tr>
<td>NMAPRSV4</td>
<td>083</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Reserved for IBM's use.</td>
</tr>
<tr>
<td>NMAPRSV5</td>
<td>084</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Reserved for IBM's use.</td>
</tr>
<tr>
<td>PWDENV</td>
<td>085</td>
<td>00</td>
<td>08</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
<td>Internal form of the enveloped RACF password.</td>
</tr>
<tr>
<td>PASSASIS</td>
<td>086</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
<td>Identifies the user as having (bit 0 is on) or not having used a mixed case password.</td>
</tr>
<tr>
<td>PHRASE</td>
<td>087</td>
<td>04</td>
<td>80</td>
<td>00000000</td>
<td>FF</td>
<td>BIN</td>
<td>The password phrase associated with this user.</td>
</tr>
<tr>
<td>PHRDATE</td>
<td>088</td>
<td>00</td>
<td>20</td>
<td>00000003</td>
<td>FF</td>
<td>BIN</td>
<td>The date the password phrase was last changed.</td>
</tr>
<tr>
<td>PHRGEN</td>
<td>089</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>FF</td>
<td>INT</td>
<td>Current password phrase generation number.</td>
</tr>
<tr>
<td>PHRCNT</td>
<td>090</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>INT</td>
<td>Number of old password phrases.</td>
</tr>
<tr>
<td>OLDPHRNM</td>
<td>091</td>
<td>80</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>INT</td>
<td>Generation number of password phrase.</td>
</tr>
<tr>
<td>OLDPHR</td>
<td>092</td>
<td>84</td>
<td>00</td>
<td>00000008</td>
<td>FF</td>
<td>BIN</td>
<td>Previous password phrase, truncated to 8 bytes.</td>
</tr>
<tr>
<td>CERTSEQN</td>
<td>093</td>
<td>00</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>INT</td>
<td>Sequence number that is incremented whenever a certificate for the user is added, deleted, or altered.</td>
</tr>
<tr>
<td>PPHENV</td>
<td>094</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>BIN</td>
<td>Internal form of the enveloped RACF password phrase.</td>
</tr>
<tr>
<td>DMAPCT</td>
<td>095</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Bin</td>
<td>Number of IDIDMAP Mapping Profiles that specify this user ID.</td>
</tr>
<tr>
<td>DMAPLABL</td>
<td>096</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
<td>Label associated with this mapping.</td>
</tr>
<tr>
<td>DMAPNAME</td>
<td>097</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
<td>Name of mapping profile. The names correspond to profiles in the IDIDMAP class.</td>
</tr>
<tr>
<td>DMAPRSV1</td>
<td>098</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
<td>Reserved for IBM's use.</td>
</tr>
<tr>
<td>DMAPRSV2</td>
<td>099</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
<td>Reserved for IBM's use.</td>
</tr>
</tbody>
</table>

User template

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
<th>Field being described</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field ID</td>
<td>Flag 1</td>
<td>Flag 2</td>
<td>Field length decimal</td>
<td>Default value</td>
<td>Type</td>
<td>Field being described</td>
<td></td>
</tr>
<tr>
<td>37–44</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Int</td>
<td>The time the user ID association was approved or refused.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>For the format of the time, see the TIME macro as documented in <a href="http://www.ibm.com">z/OS MVS Programming: Assembler Services Reference</a>.</td>
</tr>
<tr>
<td>45–56</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Char</td>
<td>Reserved for IBM's use.</td>
</tr>
<tr>
<td>57–64</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Char</td>
<td>The user ID that created the entry.</td>
</tr>
</tbody>
</table>
User template

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Combination field IDs</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFDATE</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>004 000 000 000 000</td>
<td>Combination.</td>
</tr>
<tr>
<td>CREADATE</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>004 000 000 000 000</td>
<td>Fields.</td>
</tr>
<tr>
<td>OWNER</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>005 000 000 000 000</td>
<td></td>
</tr>
<tr>
<td>PASSDATA</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>012 013 000 000 000</td>
<td></td>
</tr>
<tr>
<td>NAME</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>014 000 000 000 000</td>
<td></td>
</tr>
<tr>
<td>OLDPSWDs</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>026 027 000 000 000</td>
<td></td>
</tr>
<tr>
<td>LOGINFO</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>035 036 000 000 000</td>
<td></td>
</tr>
<tr>
<td>FIELD</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>038 039 040 000 000</td>
<td></td>
</tr>
<tr>
<td>USERDATA</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>046 047 048 000 000</td>
<td></td>
</tr>
<tr>
<td>CGDEFDAT</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>052 000 000 000 000</td>
<td></td>
</tr>
<tr>
<td>CGCREADT</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>052 000 000 000 000</td>
<td></td>
</tr>
<tr>
<td>CGOWNER</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>053 000 000 000 000</td>
<td></td>
</tr>
<tr>
<td>TUENTRY</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>068 069 000 000 000</td>
<td></td>
</tr>
<tr>
<td>CERTLIST</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>071 072 000 000 000</td>
<td></td>
</tr>
<tr>
<td>CERTLST2</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>071 072 073 074 000</td>
<td></td>
</tr>
<tr>
<td>CERTLST3</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>071 072 073 000 000</td>
<td></td>
</tr>
<tr>
<td>CERTSIGL</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>071 073 074 000 000</td>
<td></td>
</tr>
<tr>
<td>OLDPHRES</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>091 092 000 000 000</td>
<td></td>
</tr>
<tr>
<td>DMAPLST1</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>096 097 000 000 000</td>
<td>Combination for distributed identity.</td>
</tr>
</tbody>
</table>

Template

<table>
<thead>
<tr>
<th>Template Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>The following is the DFP segment of the USER template.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFP</td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>00</td>
<td>Start of segment fields</td>
</tr>
<tr>
<td>DATAPPL</td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Data Application; maximum length = 8</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Data Class; maximum length = 8</td>
</tr>
<tr>
<td>MGMTCLAS</td>
<td>004</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Management Class; maximum length = 8</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>005</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Storage Class; maximum length = 8</td>
</tr>
<tr>
<td>The following is the TSO segment of the USER template.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSO</td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>00</td>
<td>Start of segment fields</td>
</tr>
<tr>
<td>TACCNT</td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Default account numbers; maximum length = 40</td>
</tr>
<tr>
<td>TCOMMAND</td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Default command at logon; maximum length = 80</td>
</tr>
<tr>
<td>TDEST</td>
<td>004</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Destination identifier; maximum length = 8</td>
</tr>
<tr>
<td>THCLASS</td>
<td>005</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Default hold class; maximum length = 1</td>
</tr>
<tr>
<td>TJCLASS</td>
<td>006</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Default job class</td>
</tr>
<tr>
<td>TLPROC</td>
<td>007</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Default logon procedure; maximum length = 8</td>
</tr>
<tr>
<td>TLNSIZE</td>
<td>008</td>
<td>00</td>
<td>00</td>
<td>000000004</td>
<td>Int</td>
<td>Logon size</td>
</tr>
<tr>
<td>TMCLASS</td>
<td>009</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Default message class; maximum length = 1</td>
</tr>
<tr>
<td>TMSIZE</td>
<td>010</td>
<td>00</td>
<td>00</td>
<td>000000004</td>
<td>Int</td>
<td>Maximum region size</td>
</tr>
<tr>
<td>TOPTION</td>
<td>011</td>
<td>20</td>
<td>00</td>
<td>000000001</td>
<td>Bin</td>
<td>Default for mail notices and OIDcard</td>
</tr>
<tr>
<td>TPERFORM</td>
<td>012</td>
<td>00</td>
<td>00</td>
<td>000000004</td>
<td>Int</td>
<td>Performance group; stored as a two-byte value</td>
</tr>
<tr>
<td>TRBA</td>
<td>013</td>
<td>00</td>
<td>00</td>
<td>000000003</td>
<td>Bin</td>
<td>RBA of user's broadcast area</td>
</tr>
<tr>
<td>TCLASS</td>
<td>014</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Default sysout class</td>
</tr>
<tr>
<td>TUDATA</td>
<td>015</td>
<td>00</td>
<td>00</td>
<td>000000002</td>
<td>Bin</td>
<td>2 bytes of hex user data</td>
</tr>
<tr>
<td>TUNIT</td>
<td>016</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Default unit name; maximum length = 8</td>
</tr>
<tr>
<td>TUPDT</td>
<td>017</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Bin</td>
<td>Data from UPT control block</td>
</tr>
<tr>
<td>TSOSLABL</td>
<td>018</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Default logon SECLABEL; maximum length = 8</td>
</tr>
<tr>
<td>TCONS</td>
<td>019</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>Char</td>
<td>Consoles support</td>
</tr>
</tbody>
</table>

The following is the CICS® segment of the USER template.
### User template

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
<th>Field being described</th>
</tr>
</thead>
<tbody>
<tr>
<td>CICS</td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Start of segment fields</td>
</tr>
<tr>
<td>OPIDENT</td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000003</td>
<td>00</td>
<td>Char</td>
<td>Operator identification; 1 to 3 bytes in length</td>
</tr>
<tr>
<td>OPCLASSN</td>
<td>003</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
<td>Count of operator class values</td>
</tr>
<tr>
<td>OPCLASS</td>
<td>004</td>
<td>80</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Int</td>
<td>Operator class</td>
</tr>
<tr>
<td>OPPRTY</td>
<td>005</td>
<td>00</td>
<td>40</td>
<td>00000002</td>
<td>00</td>
<td>Int</td>
<td>Operator priority</td>
</tr>
<tr>
<td>XRFSOFF</td>
<td>006</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
<td>XRF re-signon option:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 0 on = FORCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 0 off = NOFORCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Terminal timeout value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Two 1-byte binary fields:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• First byte = hours (0–99)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Second byte = minutes (0–59)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>Special case:</strong> The following examples are handled the same way:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• First byte = 0 hours</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Second byte = 60 minutes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• First byte = 1 hours</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Second byte = 0 minutes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>The following is the LANGUAGE segment of the USER template.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Start of segment fields</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>User's primary language; 3-character code returned by the MVS message service. For more information, see z/OS MVS Programming: Assembler Services Guide</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>User's secondary language</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>The following is the OPERPARM segment of the USER template.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Start of segment fields</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>STORAGE keyword</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>AUTH keyword:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• X’8000’ = MASTER</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• X’4000’ = ALL</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• X’2000’ = SYS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• X’10000’ = IO</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• X’0800’ = CONS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• X’0400’ = INFO</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MFORM keyword:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 0 indicates T</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 1 indicates S</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 2 indicates J</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 3 indicates M</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 4 indicates X</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>LEVEL keyword:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 0 indicates R</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 1 indicates I</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 2 indicates CE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 3 indicates E</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 4 indicates IN</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 5 indicates NB</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Bit 6 indicates ALL</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Bit 6 is mutually exclusive with all other bits except Bit 5.</td>
</tr>
</tbody>
</table>
**User template**

<table>
<thead>
<tr>
<th>Template Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPERMON</td>
<td>006</td>
<td>00</td>
<td>00</td>
<td>00000002</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>OPERROUT</td>
<td>007</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>OPERLOGC</td>
<td>008</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>OPERMGID</td>
<td>009</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>OPERDOM</td>
<td>010</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>OPERKEY</td>
<td>011</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>OPERCMDS</td>
<td>012</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>OPERUD</td>
<td>013</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>OPERMCNT</td>
<td>014</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>OPERMSCP</td>
<td>015</td>
<td>80</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>OPERALTG</td>
<td>016</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>OPERAUTO</td>
<td>017</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>OPERHC</td>
<td>018</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>BIN</td>
</tr>
<tr>
<td>OPERINT</td>
<td>019</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>BIN</td>
</tr>
<tr>
<td>OPERUNKN</td>
<td>020</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>BIN</td>
</tr>
</tbody>
</table>

**Field being described**

- **OPERMON**: MONITOR keyword:
  - Bit 0 indicates JOBNAME.
  - Bit 1 indicates JOBNAMEST.
  - Bit 2 indicates SESS.
  - Bit 3 indicates SESST.
  - Bit 4 indicates STATUS.

- **OPERROUT**: ROUTCODE keyword; 16-bit length bitstring in which each bit indicates a particular ROUTCODE.

- **OPERLOGC**: LOGCMDRESP keyword.

- **OPERMGID**: MIGID keyword.

- **OPERDOM**: DOM keyword.

- **OPERKEY**: KEY keyword; maximum length = 8 (or '*').

- **OPERCMDS**: CMDSYS keyword; maximum length = 8 (or '*').

- **OPERUD**: UD keyword.

- **OPERMCNT**: MSCOPE systems

- **OPERMSCP**: MSCOPE systems

- **OPERALTG**: ALTGRP keyword

- **OPERAUTO**: AUTO keyword; X'80' indicates YES; X'40' indicates NO

- **OPERHC**: HC keyword; X'80' indicates YES; X'40' indicates NO

- **OPERINT**: INTIDS keyword; X'80' indicates YES; X'40' indicates NO

- **OPERUNKN**: UNKNIDS keyword; X'80' indicates YES; X'40' indicates NO

The following is the **WORK ATTRIBUTES** segment of the **USER** template.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field Length decimal</th>
<th>Default Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKATTR</td>
<td>001</td>
<td>00</td>
<td>80</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>WANAME</td>
<td>002</td>
<td>00</td>
<td>80</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>WABLDG</td>
<td>003</td>
<td>00</td>
<td>80</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>WADEPT</td>
<td>004</td>
<td>00</td>
<td>80</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>WAROOM</td>
<td>005</td>
<td>00</td>
<td>80</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
</tbody>
</table>

**User template**

- **WANAME**: User name for SYSOUT; maximum length = 60
- **WABLDG**: Building for delivery; maximum length = 60
- **WADEPT**: Department for delivery; maximum length = 60
- **WAROOM**: Room for delivery; maximum length = 60
### User template

<table>
<thead>
<tr>
<th>Template Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>WAADDR1</td>
<td>006</td>
<td>00</td>
<td>80</td>
<td>000000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>WAADDR2</td>
<td>007</td>
<td>00</td>
<td>80</td>
<td>000000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>WAADDR3</td>
<td>008</td>
<td>00</td>
<td>80</td>
<td>000000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>WAADDR4</td>
<td>009</td>
<td>00</td>
<td>80</td>
<td>000000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>WAACCNT</td>
<td>010</td>
<td>00</td>
<td>80</td>
<td>000000000</td>
<td>00</td>
<td>Char</td>
</tr>
</tbody>
</table>

**The following is the OMVS segment of the USER template.**

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>OMVS</td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>UID</td>
<td>002</td>
<td>00</td>
<td>10</td>
<td>000000004</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>HOME</td>
<td>004</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>PROGRAM</td>
<td>005</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>CPUTIME</td>
<td>006</td>
<td>00</td>
<td>00</td>
<td>000000004</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>ASSIZE</td>
<td>007</td>
<td>00</td>
<td>00</td>
<td>000000004</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>FILEPROC</td>
<td>008</td>
<td>00</td>
<td>00</td>
<td>000000004</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>PROCUSER</td>
<td>009</td>
<td>00</td>
<td>00</td>
<td>000000004</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>THREADS</td>
<td>010</td>
<td>00</td>
<td>00</td>
<td>000000004</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>MMAPAREA</td>
<td>011</td>
<td>00</td>
<td>00</td>
<td>000000004</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>MEMLIMIT</td>
<td>012</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>0</td>
<td>Char</td>
</tr>
<tr>
<td>SHMEMMAX</td>
<td>013</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>0</td>
<td>Char</td>
</tr>
</tbody>
</table>

**The following is the NETVIEW segment of the USER template.**

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>NETVIEW</td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>CONSNAME</td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>CTL</td>
<td>004</td>
<td>20</td>
<td>00</td>
<td>000000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>MSGRECVR</td>
<td>005</td>
<td>20</td>
<td>00</td>
<td>000000001</td>
<td>00</td>
<td>Bin</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPCLASSN</td>
<td>006</td>
<td>10</td>
<td>00</td>
<td>000000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>OPCLASS</td>
<td>007</td>
<td>80</td>
<td>40</td>
<td>000000002</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>DOMAINSN</td>
<td>008</td>
<td>10</td>
<td>00</td>
<td>000000004</td>
<td>00</td>
<td>Int</td>
</tr>
</tbody>
</table>

**Field being described**

- **OMVS**: Set of segment fields
  - **UID**: User ID; maximum length = 4
  - **HOME**: HOME Path; maximum length = 1023
  - **PROGRAM**: Initial Program; maximum length = 1023
  - **CPUTIME**: CPU Time Maximum
  - **ASSIZE**: ASSIZE Maximum
  - **FILEPROC**: FILEPROC Maximum
  - **PROCUSER**: PROCUSER Maximum
  - **THREADS**: THREADS Maximum
  - **MMAPAREA**: MMAPAREAMAX
  - **MEMLIMIT**: MEMLIMIT Maximum
  - **SHMEMMAX**: SHMEMMAX Maximum

- **NETVIEW**: Set of segment fields
  - **CONSNAME**: Default MCS console identifier; maximum length = 8
  - **CTL**: CTL keyword – Specifies whether a security check is performed when the operator logs on to Netview. Values:
    - X'00': Indicates CTL was not specified or CTL(SPECIFIC) was specified.
    - X'80': Indicates CTL(GLOBAL) was specified.
    - X'40': Indicates CTL(GENERAL) was specified.
  - **MSGRECVR**: MSGRECVR keyword
    - Value: Meaning when set
      - X'00': Indicates the operator can receive unsolicited messages that are not routed to a specific NetView operator.
      - X'80': Indicates the operator cannot receive unsolicited messages that are not routed to a specific NetView operator.
  - **OPCLASSN**: Count of operator class values.
  - **OPCLASS**: Specifies a NetView scope class for which the operator has authority. This is a 2-byte repeating field. Each member can have fixed-binary values from 1 to 2040.
  - **DOMAINSN**: The number of domains the NetView operator controls.
User template

<table>
<thead>
<tr>
<th>Template Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
<th>Field being described</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOMAINS</td>
<td>009</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Specifies the identifier of NetView programs in another NetView domain for which this operator has authority. This is a variable length (5-character maximum) repeating field.</td>
</tr>
<tr>
<td>NGMFADMN</td>
<td>010</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
<td>NGMFADMN keyword</td>
</tr>
<tr>
<td>NGSFMVSN</td>
<td>011</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>NetView Graphic Monitor Facility view span options; maximum length = 8</td>
</tr>
</tbody>
</table>

The following is the DCE segment of the USER template.

DCE 001 00 00 00000000 00 Char Start of segment fields
UUID 002 00 00 00000036 FF Char User's DCE principal's UUID; exactly 36 characters, in the format "nnnnnnnn-nnnn-nnnn-nnnn-nnnnnnnnnnnn" where n is any hexadecimal digit.

DCENAME 003 00 00 00000000 00 Char User's DCE principal name; maximum length = 1023
HOMECCELL 004 00 00 00000000 00 Char Home cell for this DCE user; maximum length = 1023, and it must start with either "/.../" or "/.:/
HOMEUUID 005 00 00 00000036 FF Char Home cell UUID; exactly 36 characters, in the format "nnnnnnnn-nnnn-nnnn-nnnn-nnnnnnnnnnnn" where n is any hexadecimal digit.

DCEFLAGS 006 20 00 00000001 00 Bin User flags
DPASSWD 007 00 00 00000000 00 Char Current DCE password
DCEENCRY 008 00 00 00000071 00 Bin PW mask/encrypt key

The following is the OVM segment of the USER template.

OVM 001 00 00 00000000 00 Start of segment fields
UID 002 00 00 00000004 FF Int OVM - UID
HOME 003 00 00 00000000 00 Char Home path; maximum length = 1023
PROGRAM 004 00 00 00000000 00 Char Initial program; maximum length = 1023
FSROOT 005 00 00 00000000 00 Char File system root; maximum length = 1023

The following is the LNOTES segment of the USER template.

LNOTES 001 00 00 00000000 00 Start of segment fields
SNAME 002 00 14 00000000 00 Char User's short name; maximum length = 64

The following is the NDS segment of the USER template.

NDS 001 00 00 00000000 00 Start of segment fields
UNAME 002 00 14 00000000 00 Char User's user name; maximum length = 246

The following is the KERB segment of the USER template.

KERB 001 00 00 00000000 00 Start of segment fields
KERBNAME 002 00 00 00000000 00 Char Kerberos principal name
MINTKTLF 003 00 00 00000000 00 Char Reserved for IBM's use.
MAXTKTLF 004 00 00 00000000 00 Char Maximum ticket life
DEFTKTLF 005 00 00 00000000 00 Char Reserved for IBM's use.
SALT 006 00 00 00000000 00 Char Current key salt
ENCNAME 007 00 00 00000000 00 Char Encryption type
CURKEY 008 00 00 00000000 00 Char Current key version
PREVKEY 009 00 00 00000000 00 Char Current DES key
PREVKEY 010 00 00 00000000 00 Char Previous key version
### Connect template for the RACF database

The connect template is included to maintain compatibility with previous releases. You can continue to code macros to manipulate CONNECT data. This template is provided to show what fields continue to be supported. Information that was formerly stored in CONNECT profiles was moved to the USER profile. The information is in the CGGRPCT repeat group, and the fields are prefixed by “CG”.

**Note:**

1. Application developers should not depend on being able to use RACROUTE REQUEST=EXTRACT for the BASE segment fields on any security product other than RACF. These products are expected to support only such segments as DFP and TSO.
2. The default values for this template are in the user template.

The contents of the connect template are as follows:

```plaintext
Template Field name (character data) Field ID Flag 1 Flag 2 Field length decimal Default value Type
PREVKEY 011 00 00 00000000 00 Char Previous DES key
ENCRYPT 012 00 00 00000004 55 Bin Encryption type
KEYFROM 013 00 00 00000000 00 Char Key source
  0 = PASSWORD
  1 = PHRASE

The following is the PROXY segment of the USER template.
PROXY 001 00 00 00000000 00 Start of segment fields
LDAPHOST 002 00 00 00000000 00 Char LDAP server URL; maximum length: 1023
BINDDN 003 00 00 00000000 00 Char Bind distinguished name; maximum length: 1023
BINDPW 004 00 08 00000000 00 Char Bind password; maximum length: 128
BINDPWKY 005 00 08 0000071 00 Char Bind password mask or encrypt key

The following is the EIM segment of the USER template.
EIM 001 00 00 00000000 00 Char LDAPBIND profile name

The following is the CSDATA segment of the USER template.
CSDATA 001 00 00 00 00 Start of segment fields for custom fields
CSCNT 002 10 00 4 00 Integer Count of custom fields
CSTYPE 003 80 00 1 01 Bin Custom field type:
  01 - character
  02 - numeric
  03 - flag
  04 - hex
CSKEY 004 80 00 00 00 Char Custom field keyword; maximum length = 8
CSVALUE 005 80 00 0 Char Custom field value

Field Field ID Flag 1 Flag 2 Combination field IDs Type
The following is a COMBINATION field of the CSDATA segment of the USER template.
CSCDATA 000 40 00 003 004 005 000 000 Char Combination field for custom fields
```
### Connect template

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONNECT</td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td></td>
<td>Int</td>
</tr>
<tr>
<td>ENTYPE</td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td></td>
<td>Int</td>
</tr>
<tr>
<td>VERSION</td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td></td>
<td>Int</td>
</tr>
<tr>
<td>AUTHDATE</td>
<td>004</td>
<td>00</td>
<td>A0</td>
<td>00000003</td>
<td></td>
<td>Date</td>
</tr>
<tr>
<td>AUTHOR</td>
<td>005</td>
<td>00</td>
<td>80</td>
<td>00000008</td>
<td></td>
<td>Char</td>
</tr>
<tr>
<td>LJTIME</td>
<td>006</td>
<td>01</td>
<td>00</td>
<td>00000004</td>
<td></td>
<td>Time</td>
</tr>
<tr>
<td>LJDATE</td>
<td>007</td>
<td>01</td>
<td>20</td>
<td>00000003</td>
<td></td>
<td>Date</td>
</tr>
<tr>
<td>UACC</td>
<td>008</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td></td>
<td>Bin</td>
</tr>
<tr>
<td>INITCNT</td>
<td>009</td>
<td>01</td>
<td>00</td>
<td>00000002</td>
<td></td>
<td>Int</td>
</tr>
<tr>
<td>FLAG1</td>
<td>010</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td></td>
<td>Bin</td>
</tr>
<tr>
<td>FLAG2</td>
<td>011</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td></td>
<td>Bin</td>
</tr>
<tr>
<td>FLAG3</td>
<td>012</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td></td>
<td>Bin</td>
</tr>
<tr>
<td>FLAG4</td>
<td>013</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td></td>
<td>Bin</td>
</tr>
<tr>
<td>FLAG5</td>
<td>014</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td></td>
<td>Bin</td>
</tr>
<tr>
<td>NOTRMUAC</td>
<td>015</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td></td>
<td>Bin</td>
</tr>
<tr>
<td>GRPAUDIT</td>
<td>016</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td></td>
<td>Bin</td>
</tr>
<tr>
<td>REVOKEDT</td>
<td>017</td>
<td>00</td>
<td>20</td>
<td>00000000</td>
<td></td>
<td>Date</td>
</tr>
<tr>
<td>RESUMEDT</td>
<td>018</td>
<td>00</td>
<td>20</td>
<td>00000000</td>
<td></td>
<td>Date</td>
</tr>
</tbody>
</table>

**Field being described**

- **CONNECT**: The number (3) corresponding to connect profiles.
- **ENTYPE**: The version field from the profile. Always 'X'01'.
- **AUTHDATE**: The date the user was connected to the group.
- **AUTHOR**: The owner (user ID) of the connect entry.
- **LJTIME**: The last recorded time that RACROUTE REQUEST=VERIFY was last issued for this user and group.
- **LJDATE**: The last recorded date that RACROUTE REQUEST=VERIFY was last issued for this user and group.
- **UACC**: The default universal access authority assigned to the user for this group.

**Bit Meaning when set**

- 0: ALTER access
- 1: CONTROL access
- 2: UPDATE access
- 3: READ access
- 4: EXECUTE access
- 5–6: Reserved for IBM's use
- 7: EXECUTE access

**INITCNT**: The number of RACROUTE REQUEST=VERIFY macro instructions issued for this user and group.

**FLAG1**: Identifies the user as having (bit 0 is on) or not having the ADSP attribute.

**FLAG2**: Identifies the user as having (bit 0 is on) or not having the group-SPECIAL attribute.

**FLAG3**: Identifies the user as having (bit 0 is on) or not having the group-OPERATIONS attribute.

**FLAG4**: Identifies the user as having (bit 0 is on) or not having the REVOKE attribute.

**FLAG5**: Identifies the user as having (bit 0 is on) or not having the GRPACC attribute.

**NOTRMUAC**: Identifies whether the user must be authorized by the PERMIT command with at least READ authority to access a terminal. (If not, RACF uses the terminal's universal access authority.) If bit 0 is on, the user must be specifically authorized to use the terminal.

**GRPAUDIT**: Identifies the user as having (bit 0 is on) or not having the group-AUDITOR attribute.

**REVOKEDT**: The date the user will be revoked. This field either has length 0, or contains a 3-byte revoke date.

**RESUMEDT**: The date the user will be resumed. This field either has length 0, or contains a 3-byte resume date.

---

**Field name (character data) | Field ID | Flag 1 | Flag 2 | Combination field IDs | Type**

- **DEDDATE**: 000 00 00 004 000 000 000 Char
- **CREDATE**: 000 00 00 004 000 000 000 Char

**Combination field IDs**

- **DEFDATE**: Combination.
- **CREDATE**: Fields.
Data set template for the RACF database

The data set template describes the fields of the data set profiles in a RACF database.

NOT programming interface information

ACL2VAR  CATEGORY  FLDNAME  NUMCTGY
AUDITQF   FIELD     FLDVALUE
AUDITQS   FLDCTNT

End of NOT programming interface information

Note:

1. Application developers should not depend on being able to use RACROUTE REQUEST=EXTRACT for the BASE segment fields on any security product other than RACF. These products are expected to support only such segments as DFP and TSO.

2. The TME segment fields are intended to be updated by Tivoli applications, which manage updates, permissions, and cross-references among the fields. The TME fields should only be directly updated on an exception basis. See [z/OS Security Server RACF Command Language Reference](https://www.ibm.com/support/docview.wss?uid=swg21281831) for formats of the field data as enforced by the RACF commands. Use caution when directly updating TME fields, as the updates might be overridden by subsequent actions of Tivoli applications.

The contents of the data set template are as follows:

<table>
<thead>
<tr>
<th>Template Field name</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATASET</td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>ENTYPE</td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>04</td>
<td>Int</td>
</tr>
<tr>
<td>VERSION</td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>01</td>
<td>Int</td>
</tr>
<tr>
<td>CREADATE</td>
<td>004</td>
<td>00</td>
<td>20</td>
<td>00000003</td>
<td>FF</td>
<td>Date</td>
</tr>
<tr>
<td>AUTHOR</td>
<td>005</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>FF</td>
<td>Char</td>
</tr>
<tr>
<td>LREFDAT</td>
<td>006</td>
<td>01</td>
<td>20</td>
<td>00000003</td>
<td>FF</td>
<td>Date</td>
</tr>
<tr>
<td>LCHGDAT</td>
<td>007</td>
<td>01</td>
<td>20</td>
<td>00000003</td>
<td>FF</td>
<td>Date</td>
</tr>
<tr>
<td>ACSALTR</td>
<td>008</td>
<td>01</td>
<td>00</td>
<td>00000002</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>ASCCNTL</td>
<td>009</td>
<td>01</td>
<td>00</td>
<td>00000002</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>ACSUPDT</td>
<td>010</td>
<td>01</td>
<td>00</td>
<td>00000002</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>ACSREAD</td>
<td>011</td>
<td>01</td>
<td>00</td>
<td>00000002</td>
<td>FF</td>
<td>Int</td>
</tr>
</tbody>
</table>

Field being described

- The number (4) corresponding to data set profiles.
- The version field from the profile. Always X’01’.
- The date the data set was initially defined to RACF; 3-byte date.
- The owner of the data set.
- The date the data set was last referenced; 3-byte date.
- The date the data set was last updated; 3-byte date.
- The number of times the data set was accessed with ALTER authority.
- The number of times the data set was accessed with CONTROL authority.
- The number of times the data set was accessed with UPDATE authority.
- The number of times the data set was accessed with READ authority.
Data set template

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIVACS</td>
<td>012</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
</tbody>
</table>

Field being described

The universal access authority for the data set.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ALTER access</td>
</tr>
<tr>
<td>1</td>
<td>CONTROL access</td>
</tr>
<tr>
<td>2</td>
<td>UPDATE access</td>
</tr>
<tr>
<td>3</td>
<td>READ access</td>
</tr>
<tr>
<td>4</td>
<td>EXECUTE access</td>
</tr>
<tr>
<td>5–6</td>
<td>Reserved for IBM's use</td>
</tr>
<tr>
<td>7</td>
<td>NONE access</td>
</tr>
</tbody>
</table>

| Group data set. If bit 0 is on, the data set is a group data set. Audit Flags. |

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Audit all accesses</td>
</tr>
<tr>
<td>1</td>
<td>Audit successful accesses</td>
</tr>
<tr>
<td>2</td>
<td>Audit accesses that fail</td>
</tr>
<tr>
<td>3</td>
<td>No auditing</td>
</tr>
<tr>
<td>4–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

| The current connect group of the user who created this data set. |

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>VSAM data set (non-VSAM if this bit is set to 0)</td>
</tr>
<tr>
<td>1</td>
<td>MODEL profile</td>
</tr>
<tr>
<td>2</td>
<td>Type = TAPE when set on</td>
</tr>
<tr>
<td>3–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

| Data set level. |

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Audit all accesses</td>
</tr>
<tr>
<td>1</td>
<td>Audit successful accesses</td>
</tr>
<tr>
<td>2</td>
<td>Audit accesses that fail</td>
</tr>
<tr>
<td>3</td>
<td>No auditing</td>
</tr>
<tr>
<td>4–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

| The type of device on which the data set resides; only for non-model, discrete data sets. |

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Audit all accesses</td>
</tr>
<tr>
<td>1</td>
<td>Audit successful accesses</td>
</tr>
<tr>
<td>2</td>
<td>Audit accesses that fail</td>
</tr>
<tr>
<td>3</td>
<td>No auditing</td>
</tr>
<tr>
<td>4–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

| The EBCDIC name of the device type on which the data set resides; only for non-model, discrete data sets. |

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Audit all accesses</td>
</tr>
<tr>
<td>1</td>
<td>Audit successful accesses</td>
</tr>
<tr>
<td>2</td>
<td>Audit accesses that fail</td>
</tr>
<tr>
<td>3</td>
<td>No auditing</td>
</tr>
<tr>
<td>4–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

| Global audit flags. (Audit options specified by a user with the AUDITOR or group-AUDITOR attribute.) |

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Audit all accesses</td>
</tr>
<tr>
<td>1</td>
<td>Audit successful accesses</td>
</tr>
<tr>
<td>2</td>
<td>Audit accesses that fail</td>
</tr>
<tr>
<td>3</td>
<td>No auditing</td>
</tr>
<tr>
<td>4–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

| Installation data; maximum length = 255. |

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Audit all accesses</td>
</tr>
<tr>
<td>1</td>
<td>Audit successful accesses</td>
</tr>
<tr>
<td>2</td>
<td>Audit accesses that fail</td>
</tr>
<tr>
<td>3</td>
<td>No auditing</td>
</tr>
<tr>
<td>4–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

| Global audit FAILURES qualifier. |

| The AUDITQS, AUDITQF, GAUDITQS, and GAUDITQF fields have the following format: |

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00'</td>
<td>Log access at READ level</td>
</tr>
<tr>
<td>X'01'</td>
<td>Log access at UPDATE level</td>
</tr>
<tr>
<td>X'02'</td>
<td>Log access at CONTROL level</td>
</tr>
<tr>
<td>X'03'</td>
<td>Log access at ALTER level</td>
</tr>
</tbody>
</table>
### Data set template

<table>
<thead>
<tr>
<th>Template Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUDITQS</td>
<td>022</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>FF</td>
<td>Bin</td>
</tr>
<tr>
<td>AUDITQF</td>
<td>023</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>FF</td>
<td>Bin</td>
</tr>
<tr>
<td>GAUDITQS</td>
<td>024</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>FF</td>
<td>Bin</td>
</tr>
<tr>
<td>WARNING</td>
<td>026</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>FF</td>
<td>Bin</td>
</tr>
<tr>
<td>SECLEVEL</td>
<td>027</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>NUMCTGY</td>
<td>028</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>CATEGORY</td>
<td>029</td>
<td>80</td>
<td>00</td>
<td>00000002</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>NOTIFY</td>
<td>030</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>RETPD</td>
<td>031</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>ACL2CNT</td>
<td>032</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>PROGRAM</td>
<td>033</td>
<td>80</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>USER2ACS</td>
<td>034</td>
<td>80</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>PROGACS</td>
<td>035</td>
<td>80</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>PACSCNT</td>
<td>036</td>
<td>80</td>
<td>00</td>
<td>00000002</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>ACL2VAR</td>
<td>037</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>FLDCTN</td>
<td>038</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>FLDCNT</td>
<td>039</td>
<td>80</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>FLDFLAG</td>
<td>040</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>FLDNAME</td>
<td>041</td>
<td>A0</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>VOLCNT</td>
<td>042</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>VOLSER</td>
<td>043</td>
<td>80</td>
<td>00</td>
<td>00000006</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>ACLCNT</td>
<td>044</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>USERID</td>
<td>045</td>
<td>80</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>USERACS</td>
<td>046</td>
<td>A0</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>ACSCNT</td>
<td>047</td>
<td>80</td>
<td>00</td>
<td>00000002</td>
<td>00</td>
<td>Int</td>
</tr>
</tbody>
</table>

**Field being described**

- **AUDITQS**: Audit SUCCESS qualifier.
- **AUDITQF**: Audit FAILURES qualifier.
- **GAUDITQS**: Global audit SUCCESS qualifier.
- **WARNING**: Identifies the data set as having (bit 7 is on) or not having the WARNING attribute.
- **SECLEVEL**: Data set security level.
- **NUMCTGY**: The number of categories.
- **CATEGORY**: A list of numbers corresponding to the categories to which this data set belongs.
- **NOTIFY**: User to be notified when access violations occur against a data set protected by this profile.
- **RETPD**: The number of days protection is provided for the data set. If used, the field will be a two-byte binary number.
- **ACL2CNT**: The number of program and user combinations currently authorized to access the data set.
- **PROGRAM**: The name of a program currently authorized to access the data set, or a 1-byte flag followed by 7 bytes reserved for IBM's use.
- **USER2ACS**: User ID or group.
- **PROGACS**: The access authority of the program and user combinations.
- **PACSCNT**: Access count.
- **ACL2VAR**: Additional conditional data, 9-byte length, in which the 1st byte tells what kind of access is allowed and the remaining 8 bytes contain the data.
- **FLDCNT**: Reserved for IBM's use.
- **FLDNAME**: Reserved for IBM's use.
- **FLDFLAG**: Reserved for IBM's use.
- **VOLCNT**: The number of volumes containing the data set.
- **VOLSER**: A list of the serial numbers of the volumes containing the data set.
- **ACLCNT**: The number of users and groups currently authorized to access the data set.
- **USERID**: The user ID or group name of each user or group authorized to access the data set.
- **USERACS**: The access authority that each user or group has for the data set.

**Bit Meaning when set**

- 0: ALTER access
- 1: CONTROL access
- 2: UPDATE access
- 3: READ access
- 4: EXECUTE access
- 5–6: Reserved for IBM's use
- 7: NONE access

The number of times the data set was accessed by each user or group.
### Data set template

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>USRCNT</td>
<td>048</td>
<td>10</td>
<td>00</td>
<td>000000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>USRNM</td>
<td>049</td>
<td>80</td>
<td>00</td>
<td>000000008</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>USRDATA</td>
<td>050</td>
<td>80</td>
<td>00</td>
<td>000000000</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>USRFLG</td>
<td>051</td>
<td>A0</td>
<td>00</td>
<td>000000001</td>
<td>00</td>
<td>Int</td>
</tr>
</tbody>
</table>

**SECLABEL**

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>052</td>
<td>00</td>
<td>00</td>
<td>000000008</td>
<td>00</td>
<td>Char</td>
</tr>
</tbody>
</table>

**Security label.**

The following are the **COMBINATION** fields of the data set template.

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Combination field IDs</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFDATE</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>004 000 000 000</td>
<td>Char</td>
</tr>
<tr>
<td>AUTHDATE</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>004 000 000 000</td>
<td>Char</td>
</tr>
<tr>
<td>OWNER</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>005 000 000 000</td>
<td>Char</td>
</tr>
<tr>
<td>UACC</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>012 000 000 000</td>
<td>Char</td>
</tr>
<tr>
<td>ACL2</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>033 034 035 036 037</td>
<td>Char</td>
</tr>
<tr>
<td>ACL2A3</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>033 034 035 037</td>
<td>Char</td>
</tr>
<tr>
<td>ACL2A2</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>033 034 035 036</td>
<td>Char</td>
</tr>
<tr>
<td>ACL2A1</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>033 034 035 000</td>
<td>Char</td>
</tr>
<tr>
<td>FIELD</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>039 040 041 000</td>
<td>Char</td>
</tr>
<tr>
<td>VOLUME</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>043 000 000 000</td>
<td>Char</td>
</tr>
<tr>
<td>ACL1</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>045 046 047 000</td>
<td>Char</td>
</tr>
<tr>
<td>ACL1A</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>045 046 000 000</td>
<td>Char</td>
</tr>
<tr>
<td>USERDATA</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>049 050 051 000</td>
<td>Char</td>
</tr>
</tbody>
</table>

**Template Field being described**

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td></td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>000000008</td>
<td>FF</td>
<td>Char</td>
</tr>
</tbody>
</table>

**Resource owner; must represent a user ID or group name**

**Template Field being described**

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td></td>
<td>002</td>
<td>10</td>
<td>00</td>
<td>000000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td></td>
<td>003</td>
<td>80</td>
<td>00</td>
<td>000000000</td>
<td>00</td>
<td>Char</td>
</tr>
</tbody>
</table>

**Start of segment fields**

**Count of role-access specifications**

**Role-access specifications**

**General template for the RACF database**

The general template describes the fields of general resource profiles in a RACF database.
Note:

1. Application developers should not depend on being able to use RACROUTE REQUEST=EXTRACT for the BASE segment fields on any security product other than RACF. These products are expected to support only such segments as DFP and TSO.

2. The TME segment fields are intended to be updated by Tivoli applications, which manage updates, permissions, and cross-references among the fields. The TME fields should only be directly updated on an exception basis. See z/OS Security Server RACF Command Language Reference for formats of the field data as enforced by the RACF commands. Use caution when directly updating TME fields, as the updates might be overridden by subsequent actions of Tivoli applications.

The contents of the general template are as follows:

<table>
<thead>
<tr>
<th>Template Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>GENERAL</td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>ENTYPE</td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>05</td>
<td>Int</td>
</tr>
<tr>
<td>VERSION</td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>01</td>
<td>Int</td>
</tr>
<tr>
<td>CLASTYPE</td>
<td>004</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>DEFDATE</td>
<td>005</td>
<td>00</td>
<td>20</td>
<td>00000003</td>
<td>FF</td>
<td>Date</td>
</tr>
<tr>
<td>OWNER</td>
<td>006</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>FF</td>
<td>Char</td>
</tr>
<tr>
<td>LREFDAT</td>
<td>007</td>
<td>01</td>
<td>20</td>
<td>00000003</td>
<td>FF</td>
<td>Date</td>
</tr>
<tr>
<td>LCHGDAT</td>
<td>008</td>
<td>01</td>
<td>20</td>
<td>00000003</td>
<td>FF</td>
<td>Date</td>
</tr>
<tr>
<td>ACSALTR</td>
<td>009</td>
<td>01</td>
<td>00</td>
<td>00000002</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>ASCCNTL</td>
<td>010</td>
<td>01</td>
<td>00</td>
<td>00000002</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>ACSUPDT</td>
<td>011</td>
<td>01</td>
<td>00</td>
<td>00000002</td>
<td>FF</td>
<td>Int</td>
</tr>
<tr>
<td>ACSREAD</td>
<td>012</td>
<td>01</td>
<td>00</td>
<td>00000002</td>
<td>FF</td>
<td>Int</td>
</tr>
</tbody>
</table>

The number (5) corresponding to profiles for resources defined in the class descriptor table.
The version field from the profile. Always X'01'.
The class to which the resource belongs (from the ID=class-number operand of the ICHERCDE macro).
The date the resource was defined to RACF.
The owner of the resource.
The date the resource was last referenced.
The date the resource was last updated.
The number of times the resource was accessed with ALTER authority.
The number of times the resource was accessed with CONTROL authority.
The number of times the resource was accessed with UPDATE authority.
The number of times the resource was accessed with READ authority.
### General template

<table>
<thead>
<tr>
<th>Template Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>UACC</td>
<td>013</td>
<td>20</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>AUDIT</td>
<td>014</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>LEVEL</td>
<td>015</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>GAUDIT</td>
<td>016</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>INSTDATA</td>
<td>017</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>GAUDITQF</td>
<td>021</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>FF</td>
<td>Bin</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field being described</th>
</tr>
</thead>
<tbody>
<tr>
<td>The universal access authority for the resource.</td>
</tr>
</tbody>
</table>

#### UACC

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ALTER access</td>
</tr>
<tr>
<td>1</td>
<td>CONTROL access</td>
</tr>
<tr>
<td>2</td>
<td>UPDATE access</td>
</tr>
<tr>
<td>3</td>
<td>READ access</td>
</tr>
<tr>
<td>4</td>
<td>EXECUTE access</td>
</tr>
<tr>
<td>5–6</td>
<td>Reserved for IBM's use</td>
</tr>
<tr>
<td>7</td>
<td>NONE access.</td>
</tr>
</tbody>
</table>

Audit flags.

#### AUDIT

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Audit all accesses</td>
</tr>
<tr>
<td>1</td>
<td>Audit successful accesses</td>
</tr>
<tr>
<td>2</td>
<td>Audit accesses that fail</td>
</tr>
<tr>
<td>3</td>
<td>No auditing</td>
</tr>
<tr>
<td>4–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

Resource level.

Global audit flags.

#### LEVEL

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Audit all accesses</td>
</tr>
<tr>
<td>1</td>
<td>Audit successful accesses</td>
</tr>
<tr>
<td>2</td>
<td>Audit accesses that fail</td>
</tr>
<tr>
<td>3</td>
<td>No auditing</td>
</tr>
<tr>
<td>4–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

Installation data; maximum length = 255.

Global audit FAILURES qualifier.

#### INSTDATA

The AUDITQS, AUDITQF, GAUDITQS, and GAUDITQF fields have the following format:

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00'</td>
<td>Log access at READ authority</td>
</tr>
<tr>
<td>X'01'</td>
<td>Log access at UPDATE authority</td>
</tr>
<tr>
<td>X'02'</td>
<td>Log access at CONTROL authority</td>
</tr>
<tr>
<td>X'03'</td>
<td>Log access at ALTER authority</td>
</tr>
</tbody>
</table>

Audit SUCCESS qualifier. (Audit options specified by a user with the AUDITOR or group-AUDITOR attribute.)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Audit all accesses</td>
</tr>
<tr>
<td>1</td>
<td>Audit successful accesses</td>
</tr>
<tr>
<td>2</td>
<td>Audit accesses that fail</td>
</tr>
<tr>
<td>3</td>
<td>No auditing</td>
</tr>
<tr>
<td>4–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

Audit FAILURES qualifier. (Audit options specified by a user with the AUDITOR or group-AUDITOR attribute.)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Audit all accesses</td>
</tr>
<tr>
<td>1</td>
<td>Audit successful accesses</td>
</tr>
<tr>
<td>2</td>
<td>Audit accesses that fail</td>
</tr>
<tr>
<td>3</td>
<td>No auditing</td>
</tr>
<tr>
<td>4–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>
### General template

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length (decimal)</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAUDITQS</td>
<td>020</td>
<td>00</td>
<td>00</td>
<td>00000001</td>
<td>FF</td>
<td>Bin</td>
</tr>
<tr>
<td>WARNING</td>
<td>022</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>RESFLG</td>
<td>023</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field being described</th>
</tr>
</thead>
</table>

Global audit SUCCESS qualifier. (Audit options specified by a user with the AUDITOR or group-AUDITOR attribute.)

**Bit** | **Meaning when set** |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Audit all accesses</td>
</tr>
<tr>
<td>1</td>
<td>Audit successful accesses</td>
</tr>
<tr>
<td>2</td>
<td>Audit accesses that fail</td>
</tr>
<tr>
<td>3</td>
<td>No auditing</td>
</tr>
<tr>
<td>4–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

Identifies the data set as having (bit 7 is on) or not having the WARNING attribute.

Resource profile flags:

**Bit** | **Meaning when set** |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>TAPEVOL can only contain one data set.</td>
</tr>
<tr>
<td>1</td>
<td>TAPEVOL profile is automatic.</td>
</tr>
<tr>
<td>2</td>
<td>Maintain TVTOC for TAPEVOL.</td>
</tr>
<tr>
<td>3–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

The number of TVTOC entries.

The file sequence number of tape data set.

The date the data set was created.

Data set profiles flag (RACF indicator bit):

**Bit** | **Meaning when set** |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Discrete data set profile exists</td>
</tr>
<tr>
<td>2–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

The RACF internal name.

This field is a list of the volumes on which the tape data set resides.

The name used when creating the tape data set; maximum length = 255.

The user to be notified when access violations occur against resource protected by this profile.

The days of the week the TERMINAL cannot be used. (Bit 0 equals Sunday, bit 1 equals Monday, and so on).

The time of the day the TERMINAL can be used.

The time zone in which the terminal is located.

Number of categories.

List of categories.

Resource security level.

Reserved for IBM's use.

Reserved for IBM's use.

Reserved for IBM's use.

Reserved for IBM's use.

Application data.

The number of members.

The resource group member. For SECLABEL class, a 4-byte SMF ID.

Number of volumes in tape volume set.

Volume serials of volumes in tape volume set.

The number of users and groups currently authorized to access the resource.

Appendix B. RACF database templates 373
### General template

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>USERID</td>
<td>048</td>
<td>80</td>
<td>80</td>
<td>000000008</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>USERACS</td>
<td>049</td>
<td>A0</td>
<td>80</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
</tbody>
</table>

**Field being described**

- **USERID**: The user ID or group name of each user or group authorized to access the resource.
- **USERACS**: The access authority that each user or group has for the resource.
  - **Bit Meaning when set**
    - 0: ALTER access
    - 1: CONTROL access
    - 2: UPDATE access
    - 3: READ access
    - 4: EXECUTE access
    - 5-6: Reserved for IBM's use
    - 7: NONE access
  - **Note**: Each of the above access authority fields has mutually exclusive bits except for EXECUTE and NONE.
- **ACSCNT**: The number of times the resource was accessed by each user or group.
- **USRCNT**, **USRNM**, **USRDATA**, **USRFLG**: Reserved for installation use.
- **SECLABEL**: Security label.
- **ACL2CNT**: Number of entries in conditional access list.
- **ACL2NAME**: 1 indicator byte; 7 bytes reserved for IBM's use.
- **ACL2UID**: User ID or group.
- **ACL2ACC**: Access authority.
- **ACL2ACNT**: Access count.
- **ACL2RSVD**: Conditional data. Reserved for IBM's use.
- **RACHLDR**: RACLIST header.
- **RACLDS**: RACLIST dataspace information.
- **FILTERCT**: Number of names that Hash to this DIGTNMAP Profile.
- **FLTRLABL**: Label associated with this DIGTNMAP Mapping (matches NMAPLABL for user named by FLTRUSER or user irmulti.)
- **FLTRSTAT**: Trust status – bit 0 on for trusted.
- **FLTRUSER**: User ID or criteria profile name.
- **FLTRNAME**: Unhashed issuer's name filter used to create this profile name, (max of 255), followed by a separator, (X'4A'), and the unhashed subject's name filter used to create this profile name (max of 255).
- **FLTRSV1**, **FLTRSV2**, **FLTRSV3**, **FLTRSV4**, **FLTRSV5**: Reserved for IBM's use.
- **RACDHDR**: CACHECLS header.
- **DIDCT**: Number of names that correspond to this IDIDMAP Profile.
- **DIDLABL**: Label associated with this IDIDMAP class profile mapping (matches DMAPLABL for user named by DIDUSER).
- **DIDUSER**: User ID.
### General template

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIDRNAME</td>
<td>078</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Registry name (max of 255).</td>
</tr>
<tr>
<td>DIDRSVD1</td>
<td>079</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Reserved for IBM's use.</td>
</tr>
<tr>
<td>DIDRSVD2</td>
<td>080</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Reserved for IBM's use.</td>
</tr>
</tbody>
</table>

### Field being described

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Combination field IDs</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREADATE</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>005</td>
<td>Combination.</td>
</tr>
<tr>
<td>AUTHDATE</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>005</td>
<td>Fields.</td>
</tr>
<tr>
<td>AUTHOR</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>006</td>
<td></td>
</tr>
<tr>
<td>TVTOC</td>
<td>000</td>
<td>48</td>
<td>00</td>
<td>025</td>
<td></td>
</tr>
<tr>
<td>000</td>
<td>40</td>
<td>00</td>
<td>030</td>
<td>000</td>
<td></td>
</tr>
<tr>
<td>LOGINFO</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>032</td>
<td></td>
</tr>
<tr>
<td>000</td>
<td>40</td>
<td>00</td>
<td>033</td>
<td>034</td>
<td></td>
</tr>
<tr>
<td>FIELD</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>039</td>
<td></td>
</tr>
<tr>
<td>000</td>
<td>40</td>
<td>00</td>
<td>040</td>
<td>041</td>
<td></td>
</tr>
<tr>
<td>ACL</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>048</td>
<td></td>
</tr>
<tr>
<td>ACL1</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>048</td>
<td></td>
</tr>
<tr>
<td>USERDATA</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>052</td>
<td>Conditional access list</td>
</tr>
<tr>
<td>ACL2</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>057</td>
<td>Conditional access list</td>
</tr>
<tr>
<td>ACL2A3</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>057</td>
<td></td>
</tr>
<tr>
<td>FLTRLS1</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>065</td>
<td>Combo field for FILTER</td>
</tr>
<tr>
<td>FLTRLS2</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>065</td>
<td>Combo field for FILTER</td>
</tr>
<tr>
<td>CERTRING</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>010</td>
<td>Digital certificate data.</td>
</tr>
<tr>
<td>CERTRNG2</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>009</td>
<td></td>
</tr>
<tr>
<td>CERTRNG3</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>012</td>
<td></td>
</tr>
<tr>
<td>DIDLIST1</td>
<td>000</td>
<td>40</td>
<td>00</td>
<td>076</td>
<td>Combination for distributed identity.</td>
</tr>
</tbody>
</table>

### Field being described

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SESSION</td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Start of segment fields</td>
</tr>
<tr>
<td>SESSKEY</td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Session key; maximum length = 8</td>
</tr>
<tr>
<td>SLSFLAGS</td>
<td>003</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Session flag byte</td>
</tr>
</tbody>
</table>

#### Bit Meaning when set

<table>
<thead>
<tr>
<th>Bit</th>
<th>Meaning when set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SLSLOCK-This profile is locked out</td>
</tr>
<tr>
<td>1–7</td>
<td>Reserved for IBM's use</td>
</tr>
</tbody>
</table>

Last date session key was changed. It is in the format 0cyyddF where c=0 for 1900–1999 and c=1 for 2000–2099. For more information on this MVS-retumed format, see [z/OS MVS Programming: Assembler Services manual](https://www.ibm.com/support/knowledgenterprise/dotini/infocenter/linux/en/index.jsp).

### Field being described

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEYDATE</td>
<td>004</td>
<td>00</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Date</td>
</tr>
<tr>
<td>KEYINTVL</td>
<td>005</td>
<td>00</td>
<td>00</td>
<td>00000002</td>
<td>00</td>
<td>Int Number of days before session key expires</td>
</tr>
<tr>
<td>SLSFAIL</td>
<td>006</td>
<td>00</td>
<td>00</td>
<td>00000002</td>
<td>00</td>
<td>Int Current number of invalid attempts</td>
</tr>
<tr>
<td>MAXFAIL</td>
<td>007</td>
<td>00</td>
<td>00</td>
<td>00000002</td>
<td>00</td>
<td>Int Number of invalid attempts before lockout</td>
</tr>
<tr>
<td>SENTCNT</td>
<td>008</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int Number of session entities in list</td>
</tr>
<tr>
<td>SENTITY</td>
<td>009</td>
<td>80</td>
<td>00</td>
<td>00000035</td>
<td>00</td>
<td>Char Entity name</td>
</tr>
<tr>
<td>SENTFLCT</td>
<td>010</td>
<td>80</td>
<td>00</td>
<td>00000002</td>
<td>00</td>
<td>Int Number of failed attempts for this entity</td>
</tr>
</tbody>
</table>
General template

<table>
<thead>
<tr>
<th>Template Field name</th>
<th>Character data</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONVSEC</td>
<td></td>
<td>011</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
</tbody>
</table>

The following is the DLFDATA segment of the GENERAL template.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Character data</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLFDATA</td>
<td></td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>RETAIN</td>
<td></td>
<td>002</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>JOBNMCNT</td>
<td></td>
<td>003</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>JOB NAMES</td>
<td></td>
<td>004</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
</tbody>
</table>

The following is the SSIGNON segment of the GENERAL template.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Character data</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSIGNON</td>
<td></td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>SSKEY</td>
<td></td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
</tr>
</tbody>
</table>

The following is the STDATA segment of the GENERAL template.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Character data</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>STDATA</td>
<td></td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>STUSER</td>
<td></td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>40</td>
<td>Char</td>
</tr>
<tr>
<td>STGROUP</td>
<td></td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>40</td>
<td>Char</td>
</tr>
<tr>
<td>FLAGTRUS</td>
<td></td>
<td>004</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>FLAGPRIV</td>
<td></td>
<td>005</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>FLAGTRAC</td>
<td></td>
<td>006</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
</tbody>
</table>

The following is the SVFMR segment of the GENERAL template.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Character data</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVFMR</td>
<td></td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>SCRIPTN</td>
<td></td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>PARMN</td>
<td></td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
</tbody>
</table>

The following is the CERTDATA segment of the GENERAL template.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Character data</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERTDATA</td>
<td></td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>CERT</td>
<td></td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>CERTPRVK</td>
<td></td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>RINGCT</td>
<td></td>
<td>004</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>RINGNAME</td>
<td></td>
<td>005</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>CERTSTRT</td>
<td></td>
<td>006</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>CERTEND</td>
<td></td>
<td>007</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>CERTCT</td>
<td></td>
<td>008</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>CERTNAME</td>
<td></td>
<td>009</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
</tbody>
</table>

Field being described

Conversation security.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'40'</td>
<td>Conversation security</td>
</tr>
<tr>
<td>X'50'</td>
<td>Persistent verification</td>
</tr>
<tr>
<td>X'60'</td>
<td>User ID and password already</td>
</tr>
<tr>
<td></td>
<td>verified</td>
</tr>
<tr>
<td>X'70'</td>
<td>User ID and password already</td>
</tr>
<tr>
<td></td>
<td>verified plus persistent</td>
</tr>
<tr>
<td></td>
<td>verification</td>
</tr>
<tr>
<td>X'80'</td>
<td>Security none</td>
</tr>
</tbody>
</table>

The following is the SSIGNON segment of the GENERAL template.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Character data</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSIGNON</td>
<td></td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>SSKEY</td>
<td></td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
</tr>
</tbody>
</table>

The following is the STDATA segment of the GENERAL template.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Character data</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>STDATA</td>
<td></td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>STUSER</td>
<td></td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>40</td>
<td>Char</td>
</tr>
<tr>
<td>STGROUP</td>
<td></td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>40</td>
<td>Char</td>
</tr>
<tr>
<td>FLAGTRUS</td>
<td></td>
<td>004</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>FLAGPRIV</td>
<td></td>
<td>005</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>FLAGTRAC</td>
<td></td>
<td>006</td>
<td>20</td>
<td>00</td>
<td>00000001</td>
<td>00</td>
<td>Bin</td>
</tr>
</tbody>
</table>

The following is the SVFMR segment of the GENERAL template.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Character data</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVFMR</td>
<td></td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>SCRIPTN</td>
<td></td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>PARMN</td>
<td></td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>00000008</td>
<td>00</td>
<td>Char</td>
</tr>
</tbody>
</table>

The following is the CERTDATA segment of the GENERAL template.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Character data</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERTDATA</td>
<td></td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>CERT</td>
<td></td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>CERTPRVK</td>
<td></td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Bin</td>
</tr>
<tr>
<td>RINGCT</td>
<td></td>
<td>004</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>RINGNAME</td>
<td></td>
<td>005</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
<tr>
<td>CERTSTRT</td>
<td></td>
<td>006</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>CERTEND</td>
<td></td>
<td>007</td>
<td>00</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>CERTCT</td>
<td></td>
<td>008</td>
<td>10</td>
<td>00</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
</tr>
<tr>
<td>CERTNAME</td>
<td></td>
<td>009</td>
<td>80</td>
<td>00</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
</tr>
</tbody>
</table>

The profile name of the certificate
### General template

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
<th>Field being described</th>
</tr>
</thead>
</table>
| CERTUSAG                   | 010      | 80     | 00     | 00000004             | 00            | Bin  | Certificate usage in ring:  
|                            |          |        |        |                      |               |      | • X'00000000' – PERSONAL  
|                            |          |        |        |                      |               |      | • X'00000001’ – SITE  
|                            |          |        |        |                      |               |      | • X'00000002' – CERTAUTH  
| CERTDFLT                   | 011      | 80     | 00     | 00000001             | 00            | Bin  | Verifies if it is the default certificate:  
|                            |          |        |        |                      |               |      | • X'00' – Not the default  
|                            |          |        |        |                      |               |      | • X'80' – The default  
| CERTSJDN                   | 012      | 80     | 00     | 00000000             | 00            | Bin  | The subject name of the entity to whom the certificate is issued. This field is a BER-encoded format of the subject’s distinguished name as contained in the certificate  
| CERTLABL                   | 013      | 80     | 00     | 00000000             | 00            | Char | Label associated with the certificate  
| CERTRSV1                   | 014      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSV2                   | 015      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSV3                   | 016      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSV4                   | 017      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSV5                   | 018      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSV6                   | 019      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSV7                   | 020      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSV8                   | 021      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSV9                   | 022      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSVA                   | 023      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSVB                   | 024      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSVC                   | 025      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSVD                   | 026      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSVE                   | 027      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSVF                   | 028      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSVG                   | 029      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSVH                   | 030      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSVI                   | 031      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSVJ                   | 032      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTRSVK                   | 033      | 80     | 00     | 00000000             | 00            | Bin  | Reserved for IBM's use.  
| CERTPRVT                   | 034      | 00     | 00     | 00000004             | 00            | Bin  | Associated key type:  
|                            |          |        |        |                      |               |      | • X'000000000' – No associated key  
|                            |          |        |        |                      |               |      | • X'00000001' – PKCS DER-encoded  
|                            |          |        |        |                      |               |      | • X'00000002' – ICSF token label  
|                            |          |        |        |                      |               |      | • X'00000003' – PCICC label  
|                            |          |        |        |                      |               |      | • X'00000004' – DSA  
|                            |          |        |        |                      |               |      | • X'00000005' – ICSF public token label  
|                            |          |        |        |                      |               |      | • X'00000006' – Reserved for IBM's use  
|                            |          |        |        |                      |               |      | • X'00000007' – NIST ECC key  
|                            |          |        |        |                      |               |      | • X'00000008' – Brainpool ECC key  
|                            |          |        |        |                      |               |      | • X'00000009' – NIST ECC token label in PKDS  
|                            |          |        |        |                      |               |      | • X'0000000A' – Brainpool ECC token label in PKDS  
|                            |          |        |        |                      |               |      | • X'0000000B' – RSA token label in TKDS  
|                            |          |        |        |                      |               |      | • X'0000000C' – NIST ECC token label in TKDS  
|                            |          |        |        |                      |               |      | • X'0000000D' – Brainpool ECC token label in TKDS  
| CERTPRVS                   | 035      | 00     | 00     | 00000004             | 00            | Int  | Private key size in bits  
| CERTTLSER                  | 036      | 00     | 00     | 00000008             | 00            | Bin  | The low order 8 bytes of the last certificate that was signed with this key. This field is used with DIGTCERT profiles only  
| RINGSEQN                   | 037      | 00     | 00     | 00000004             | 00            | Int  | Ring change count  

The following is the TME segment of the GENERAL template.

TME 001 00 00 00000000 00
**General template**

<table>
<thead>
<tr>
<th>Template</th>
<th>Field name (character data)</th>
<th>Field length (decimal)</th>
<th>Default value</th>
<th>Type</th>
<th>Field being described</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PARENT</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Parent name</td>
</tr>
<tr>
<td></td>
<td>CHILDN</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
<td>Count of children</td>
</tr>
<tr>
<td></td>
<td>CHILDREN</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Child names</td>
</tr>
<tr>
<td></td>
<td>RESN</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
<td>Count of resource-access specifications</td>
</tr>
<tr>
<td></td>
<td>RESOURCE</td>
<td>00000000</td>
<td>00</td>
<td>Int</td>
<td>Resource-access specifications</td>
</tr>
<tr>
<td></td>
<td>GROUPN</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
<td>Count of groups</td>
</tr>
<tr>
<td></td>
<td>GROUPS</td>
<td>00000008</td>
<td>00</td>
<td>Group names</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ROLEN</td>
<td>00000004</td>
<td>00</td>
<td>Int</td>
<td>Count of role-access specifications</td>
</tr>
<tr>
<td></td>
<td>ROLES</td>
<td>00000000</td>
<td>00</td>
<td>Char</td>
<td>Role-access specifications</td>
</tr>
</tbody>
</table>

The following is the **KERB** segment of the **GENERAL** template.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>KERB</td>
<td>001</td>
</tr>
<tr>
<td>KERBNAME</td>
<td>002</td>
</tr>
<tr>
<td>MINTKTLF</td>
<td>003</td>
</tr>
<tr>
<td>MAXTTLF</td>
<td>004</td>
</tr>
<tr>
<td>DEFTTLF</td>
<td>005</td>
</tr>
<tr>
<td>SALT</td>
<td>006</td>
</tr>
<tr>
<td>ENCTYPE</td>
<td>007</td>
</tr>
<tr>
<td>CURKEYV</td>
<td>008</td>
</tr>
<tr>
<td>CURKEY</td>
<td>009</td>
</tr>
<tr>
<td>PREVKEYV</td>
<td>010</td>
</tr>
<tr>
<td>PREVKEY</td>
<td>011</td>
</tr>
<tr>
<td>ENCRYPT</td>
<td>012</td>
</tr>
<tr>
<td>CHKADDRS</td>
<td>013</td>
</tr>
</tbody>
</table>

Start of segment fields

Kerberos realm name

Minimum ticket life

Maximum ticket life

Default ticket life

Current key salt

Encryption type

Current key version

Current DES key

Previous key version

Previous DES key

Encryption type

Check addresses flag

The following is the **PROXY** segment of the **GENERAL** template.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROXY</td>
<td>001</td>
</tr>
<tr>
<td>LDAPHOST</td>
<td>002</td>
</tr>
<tr>
<td>BINDDN</td>
<td>003</td>
</tr>
<tr>
<td>BINDPW</td>
<td>004</td>
</tr>
<tr>
<td>BINDPWKY</td>
<td>005</td>
</tr>
</tbody>
</table>

LDAP server URL; maximum length: 1023

Bind distinguished name; maximum length: 1023

Bind password; maximum length: 128

Bind password mask or encrypt key

Start of segment fields

The following is the **EIM** segment of the **GENERAL** template.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>EIM</td>
<td>001</td>
</tr>
<tr>
<td>DOMAINDN</td>
<td>002</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>003</td>
</tr>
<tr>
<td>LOCALREG</td>
<td>004</td>
</tr>
<tr>
<td>KERBREG</td>
<td>005</td>
</tr>
<tr>
<td>X509REG</td>
<td>006</td>
</tr>
</tbody>
</table>

EIM Domain Distinguished Names

EIM Options

Local Registry Name

Kerberos Registry Name

X509 Registry Name

Start of segment fields

The following is the **ALIAS** segment of the **GENERAL** template.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALIAS</td>
<td>001</td>
</tr>
<tr>
<td>ILOOK</td>
<td>002</td>
</tr>
</tbody>
</table>

Start of segment fields

IP lookup value

Start of segment fields

The following is the **CDTINFO** segment of the **GENERAL** template.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDTINFO</td>
<td>001</td>
</tr>
<tr>
<td>CDTPOSIT</td>
<td>002</td>
</tr>
<tr>
<td>CDTMAXLN</td>
<td>003</td>
</tr>
<tr>
<td>CDTMAXLX</td>
<td>004</td>
</tr>
<tr>
<td>CDTDFTRC</td>
<td>005</td>
</tr>
<tr>
<td>CDTKEYQL</td>
<td>006</td>
</tr>
<tr>
<td>CDTGROUP</td>
<td>007</td>
</tr>
<tr>
<td>CDTMEMBR</td>
<td>008</td>
</tr>
</tbody>
</table>

Start of segment fields

POSIT number for class

Maximum length of profile names

Maximum resource or profile name length when using ENTITYX

Default return code

Number of key qualifiers

Resource grouping class name

Member class name

Start of segment fields
### General template

<table>
<thead>
<tr>
<th>Template Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
<th>Field being described</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDTFIRST</td>
<td>009</td>
<td>00</td>
<td>00</td>
<td>1</td>
<td>X'C0'</td>
<td>Bin</td>
<td>Character restriction for first character of profile name</td>
</tr>
</tbody>
</table>

**Value** | **Meaning**
---|---
X'80' | Alphabetic
X'40' | National
X'20' | Numeric
X'10' | Special

| CDTOTHER                           | 010      | 00     | 00     | 1                     | X'C0'        | Bin  | Character restriction for characters of the profile name other than the first character |

**Value** | **Meaning**
---|---
X'80' | Alphabetic
X'40' | National
X'20' | Numeric
X'10' | Special

| CDOTPER                            | 011      | 00     | 00     | 1                     | X'00'        | Bin  | Operations attribute considered |

**Value** | **Meaning**
---|---
X'80' | RACF considers OPERATIONS attribute

**Default UACC**

| CDTUACC                            | 012      | 00     | 00     | 1                     | X'01'        | Bin  | Profiles allowed |

**Value** | **Meaning**
---|---
X'80' | Profiles are allowed

| CDTRACL                            | 013      | 00     | 00     | 1                     | X'00'        | Bin  | Security labels required |

**Value** | **Meaning**
---|---
X'80' | Security labels are required

| CDTGENL                            | 014      | 00     | 00     | 1                     | X'00'        | Bin  | Mandatory access checking (MAC) processing |

**Value** | **Meaning**
---|---
X'80' | Normal mandatory access checks
X'40' | Reverse mandatory access checks
X'20' | Equal mandatory access checks

| CDTMAC                              | 015      | 00     | 00     | 1                     | X'80'        | Bin  | ENF Signal |

**Value** | **Meaning**
---|---
X'80' | ENF signal to be sent

| CDTSIGL                             | 016      | 00     | 00     | 1                     | X'00'        | Bin  | Case of profile names |

**Value** | **Meaning**
---|---
X'00' | Uppercase
X'80' | ASIS - preserve case
General template

<table>
<thead>
<tr>
<th>Template</th>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDTGEN</td>
<td></td>
<td>020</td>
<td>00</td>
<td>00</td>
<td>1</td>
<td>X'80'</td>
<td>Bin</td>
</tr>
</tbody>
</table>

The following is the ICTX segment of the GENERAL template.

| ICTX     |                             | 001      | 00     | 00     | 00000000             | 00           | Bin  |
| USEMAP   |                             | 002      | 00     | 00     | 00000001             | 80           | Bin  |
| DOMAP    |                             | 003      | 00     | 00     | 00000001             | 00           | Bin  |
| MAPREQ   |                             | 004      | 00     | 00     | 00000001             | 00           | Bin  |
| MAPTIMEO |                             | 005      | 00     | 00     | 00000002             | 00           | Int  |

The following is the CFDEF segment of the GENERAL template.

| CFDEF    |                             | 001      | 00     | 00     | 00                   | 0            | Bin  |
| CFDTYPE  |                             | 002      | 00     | 00     | 01                   | 01           | Bin  |
| CFMXLEN  |                             | 003      | 00     | 00     | 4                    | FF           | Int  |
| CFMXVAL  |                             | 004      | 00     | 00     | 4                    | FF           | Int  |
| CFMNVAL  |                             | 005      | 00     | 00     | 4                    | FF           | Int  |
| CFFIRST  |                             | 006      | 00     | 00     | 1                    | 00           | Bin  |
| CFOTHER  |                             | 007      | 00     | 00     | 1                    | 00           | Bin  |
| CFMIXED  |                             | 008      | 20     | 00     | 1                    | 00           | Bin  |
| CFHELP   |                             | 009      | 00     | 00     | 00                   | 00           | Char |
| CFLIST   |                             | 010      | 00     | 00     | 00                   | 00           | Char |

The following is the SIGVER segment of the GENERAL template.

| SIGVER   |                             | 001      | 00     | 00     | 0                    | 0            | Bin  |
| SIGREQD  |                             | 002      | 00     | 00     | 1                    | 0            | Bin  |
| FAILLOAD |                             | 003      | 00     | 00     | 1                    | 0            | Bin  |

The following is the ICTX segment of the GENERAL template.

| USEMAP   |                             | 002      | 00     | 00     | 00000001             | 80           | Bin  |

Value Meaning
X'80' GENERIC allowed
Start of segment fields
Application supplied mapping
Value Meaning
X'80' Use the mapping
Identity cache mapping
Value Meaning
X'80' Do the mapping
Mapping timeout adjustment
Mapping is required
Start of segment fields for defining custom field attributes
Data type for custom field:
* 01 - character
* 02 - numeric
* 03 - flag
* 04 - hex
Maximum field length
Maximum numeric value
Minimum numeric value
First character restrictions:
* 01 - alpha
* 02 - alphanum
* 03 - any
* 04 - nonatabc
* 05 - nonatnum
* 06 - numeric
Other character restrictions:
* 01 - alpha
* 02 - alphanum
* 03 - any
* 04 - nonatabc
* 05 - nonatnum
* 06 - numeric
If bit 0 is on, mixed case is allowed
Help text; maximum length = 255
List heading text; maximum length = 40
Start of segment fields
Module must have a signature:
Value Meaning
X'80' Yes
X'00' No
Loader failure conditions:
Value Meaning
X'80' Bad signature only
X'40' Any failing signature condition
X'00' Never
### General template

<table>
<thead>
<tr>
<th>Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIGAUDIT</td>
<td>004</td>
<td>00</td>
<td>00</td>
<td>1</td>
<td>0</td>
<td>Bin</td>
</tr>
</tbody>
</table>

The following is the ICSF segment of the GENERAL template.

| ICSF | 01 | 00 | 00 | 000000000 | 00 |
|-----------------------------|----------|--------|--------|----------------------|---------------|------|
| CSFSEXEXP                  | 02        | 00     | 00     | 00000001       | 00             | Bin  |

| CSFSKLCT                    | 03        | 10     | 00     | 00000004       | 00            | Int  |
| CSFSKLBS                    | 04        | 80     | 00     | 00000000       | 00            | Char |
| CSFSCLCT                    | 05        | 10     | 00     | 00000004       | 0             | Int  |
| CSFSCLBS                    | 06        | 80     | 00     | 00000000       | 00            | Char |
| CSFAUSE                     | 07        | 00     | 00     | 00000004       | 55            | Bin  |

| CSFSCPW                     | 08        | 00     | 00     | 00000001       | 00            | Bin  |

### Reserved template for the RACF database

This template is reserved for IBM's use. The installation must not use it.

The contents of the reserved template are as follows:

<table>
<thead>
<tr>
<th>Template Field name (character data)</th>
<th>Field ID</th>
<th>Flag 1</th>
<th>Flag 2</th>
<th>Field length decimal</th>
<th>Default value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSVTMP03</td>
<td>001</td>
<td>00</td>
<td>00</td>
<td>000000000</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>ENTYPE</td>
<td>002</td>
<td>00</td>
<td>00</td>
<td>000000001</td>
<td>00</td>
<td></td>
</tr>
<tr>
<td>VERSION</td>
<td>003</td>
<td>00</td>
<td>00</td>
<td>000000001</td>
<td>00</td>
<td></td>
</tr>
</tbody>
</table>

The number corresponding to the type of profile being described.

Template version number.
Appendix C. System authorization facility (SAF) and SAF exits

System authorization facility (SAF)

The system authorization facility (SAF) provides an installation with centralized control over system security processing through a system service called the MVS router. The MVS router provides a focal point for all products that provide resource management. The resource management components and subsystems call the MVS router as part of security decision-making functions in their processing, such as access control checking and authorization-related checking. These functions are called “control points”. SAF supports the use of common control points across products and across systems.

To use the MVS router, a resource management component or subsystem issues the RACROUTE macro. The RACROUTE macro accepts all valid parameters for any of the independent RACF system macros (RACDEF, RACINIT, RACHECK, RACLST, RACXTRT, and FRACHECK). RACROUTE verifies that only valid parameters have been coded and then passes the parameters to the MVS router.

The RACROUTE macro invokes the MVS router. When it is invoked, the MVS router first calls an optional installation exit routine and then calls the external security product (such as RACF), if one is active and installed on the system, as shown in the illustration that follows.

If an external security product is not available, you can use the MVS router exit as an installation-written security processing (or routing) routine. If an external security product is available, you can use the MVS router exit as a preprocessing exit routine for the security product. The MVS router exit routine is ICHRTX00.

After system initialization is complete, ICHRTX00 receives control for all subsequent requests for the duration of the IPL. See “Programming considerations” on page 384 for information on coding ICHRTX00.

Exit routine environment

ICHRTX00 receives control in the following environment:
- Is entered through a branch and link macro. Therefore, the exit routine runs in the same key and state as the issuer of the RACROUTE macro.
ICHRTX00

- Enabled for interrupts.
- Must be link-edited with AMODE(ANY) and RMODE(24).
- With no locks held, except the LOCAL lock can be held when
  REQUEST=FASTAUTH, and CLASS=PROGRAM, REQSTOR=PADSCHK and
  SUBSYS=CONTENTS are specified. When called with the LOCAL lock held, the
  exit must not release the lock.
- Caller's address space.
- Can be invoked in SRB mode. If the routine is invoked in SRB mode, the exit
  routine must follow SRB conventions.

An installation must provide its own recovery routine for ICHRTX00. If the exit
routine terminates abnormally, the recovery routine gets control first.

Exit routine processing

Normally, a caller invokes the MVS router and passes it class, requester, and
subsystem parameters through the RACROUTE parameter list. Using those
parameters, the MVS router invokes ICHRTX00. ICHRTX00 returns to the MVS
router with a return code that indicates whether further security processing is to
occur.

If the return code is 0, the MVS router invokes the external security product by
calling its router, ICHRFR00. ICHRFR00 then invokes the other external security
product processing and reports the results of that invocation to the MVS router by
placing a return code in register 15 and the detailed RACF-compatible return and
reason codes in the first and second words (respectively) of the RACROUTE
parameter list. For more information on the return codes the exit routine can set,
see “Return codes from the SAF router exits (ICHRTX00 and ICHRTX01)” on page
387.

If automatic direction of application updates is active, some macro requests are
propagated to other systems. If this is the case at your installation, you might need
to coordinate exit processing to avoid duplicate update requests. If a RACROUTE
was propagated, the exit might need to avoid propagating the request if it was
already propagated by the originating system's exit. Check the propagation bit in
the associated RACROUTE parameter list and avoid duplicate update requests.

Programming considerations

ICHRTX00 must be reentrant.

In addition to the address of the RACROUTE parameter list, ICHRTX00 also
receives the address of a 152-byte work area.

SAF performs functions other than being a router, such as creating security tokens
for certain RACROUTE request types, propagating user IDs, and creating default
control blocks (ACEEs) when an external security product is not available to the
system. IBM recommends that, in coding ICHRTX00, you do not bypass these SAF
functions. SAF creates and returns control blocks (tokens or ACEEs) whenever the
following RACROUTE request types are issued:

- REQUEST=VERIFYX
- REQUEST=TOKENMAP
- REQUEST=TOKENXTR
- REQUEST=TOKENBLD
SAF also creates default ACEEs for REQUEST=VERIFY when an external security product is not available on the system. System code, such as JES, requires these control blocks. Therefore, if your ICHRTX00 exit routine bypasses SAF security functions, your installation must construct and return the control blocks that SAF would have created. If you do not provide the required control blocks, problems can result. The token fields are mapped by macro ICHRUTKN (data area RUTKN). See z/OS Security Server RACF Data Areas for a mapping of the RUTKN data area.

SAF router exit

The system authorization facility (SAF) and the SAF router are present on all MVS systems, even if RACF is not installed. Although the SAF router is not part of RACF, many system components and programs invokes RACF through the RACROUTE macro and SAF. Therefore, installations can modify RACF parameter lists and do customized security processing within the SAF router.

Depending on the level of JES programming support available on your system, JES can invoke the system authorization facility (SAF) after it reads a job. JES issues the RACROUTE macro with the REQUEST=VERIFY and ENVIR=VERIFY parameters. This invocation of SAF does not result in a call to RACF; it provides an opportunity for the installation to install its own user ID verification with the SAF router exit.

The router exit must have RMODE(24) and AMODE(ANY).

SAF exits (ICHRTX00 and ICHRTX01)

The system authorization facility (SAF) provides an installation with centralized control over system security processing by using a system service called the SAF router. It is not a part of the RACF program product. The SAF router provides a focal point and a common system interface for all products providing resource control.

Considerations

The resource-managing components and subsystems call the SAF router as part of certain decision-making functions in their processing, such as access-control checking and authorization-related checking. These functions are called “control points.” This single SAF interface encourages the sharing of common control functions across products and across systems.

The SAF router is always present, whether RACF is present or not. If RACF is available on the system, the SAF router can pass control to the RACF router (ICHRFR00) that invokes the appropriate RACF function. The parameter information and the RACF router table (module ICHRFR01, if it exists) which associates router invocations with RACF functions, determine the appropriate function. Before the RACF router is called, the SAF router calls one of the optional, installation-supplied, security-processing exits if one has been installed.

When a program invokes the RACROUTE macro, the SAF router calls one of the two router exits, ICHRTX00 or ICHRTX01. If the installation has not supplied the exits or if the exit return code indicates a call to RACF, the SAF router calls the RACF router to process the request.
Control points that issue the RACROUTE macro instruction enter the SAF router in the same key and state as the RACROUTE issuer. Control points that continue to issue the independent RACF system macros (such as RACHECK and RACDEF) go directly to RACF, bypassing the router.

Special care must be taken with the SAF router exits ICHRTX00 and ICHRTX01. These exits can be invoked by callers running in any key or system mode, and running in either 31-bit or 24-bit addressing mode. These exits must be capable of running in either 31-bit or 24-bit addressing mode, and must be marked AMODE(ANY), RMODE(24). ICHRTX00 can be invoked by callers in SRB mode or cross-memory mode and can be invoked with the LOCAL lock held; be sure to consider these factors if you use this exit. ICHRTX01 is invoked early (at MVS initialization), which is a constantly changing environment. It is recommended that you not use this exit.

SAF router exits (ICHRTX00 and ICHRTX01)

The SAF router provides two optional installation exits, which are invoked whether or not RACF is installed and active on the system. If RACF is not available, the router exits might act as an installation-written security-processing (or routing) routine. If RACF is available, the exits act as a RACF preprocessing exit.

SAF initialization occurs early in the IPL, during the nucleus-initialization program (NIP). SAF exit ICHRTX01 is the exit invoked starting at SAF initialization but before master-scheduler initialization (MSI). This exit enables installations to receive control in the restricted environment available during NIP to make early security decisions. During MSI, SAF exit ICHRTX00 replaces ICHRTX01 and is the SAF exit that is invoked from then on.

The SAF router exit routines can be called only by issuing the RACROUTE macro instruction. The exits are entered by a branch and link instruction and execute in the same key and state as the issuer of the RACROUTE macro. The exits can receive parameters that point to storage in a key different from the issuer of the RACROUTE. If the caller needs to access this storage, coding of the MODESET macro to switch to the key of this storage would be required in the exits. The exits must be named ICHRTX01 or ICHRTX00, and must be in the link-pack area (LPA). The router exit must be reentrant. The router passes the parameter list to the exit routines. In addition, the exits receive the address of a 152-byte work area.

With a RACROUTE REQUEST=FASTAUTH invocation from contents supervisor to the external security manager, a local lock is held if the CLASS parameter specifies PROGRAM, the REQSTOR parameter specifies PADSCHK, and the SUBSYS parameter specifies CONTENTS. The ICHRTX00 and ICHRTX01 exits that process this call must not release this lock and must not invoke system services that would fail due to the lock being held.

The exit must be sensitive to the environment that it is called in since it runs in that same environment. For example, CICS 4.1 issues a RACROUTE REQUEST=VERIFY,ENVIR=VERIFY in a non-authorized state, so ICHRTX00 cannot issue a RACROUTE (such as a RACROUTE REQUEST=VERIFY,ENVIR=CREATE) that needs to be authorized when it is processing this call. For more information, see “Exit routine requirements” on page 388.
Control points that continue to use the independent RACF system macros (such as FRACHECK and RACDEF) do not invoke the SAF router exit routines.

When a RACF callable service for z/OS UNIX invokes RACROUTE REQUEST=FASTAUTH for a resource in the UNIXPRIV class, the SAF router exit (ICHRTX00) is not invoked.

On MVS/XA and later releases, the exit must have RMODE(24) and AMODE(ANY).

On entry to the SAF Router exit routines, register 1 points to an area containing the following addresses:

<table>
<thead>
<tr>
<th>Offset</th>
<th>Length</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4</td>
<td>Parameter-list address: Pointer to the SAF router parameter. The SAF router parameter list (mapped by macro ICHSAFP) is generated when the RACROUTE macro is issued. It describes the security-processing request by providing the request type.</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>Work-area address: Pointer to a 152-byte work area that the exits can use.</td>
</tr>
</tbody>
</table>

Return codes from the SAF router exits (ICHRTX00 and ICHRTX01)

The exit routines return one of the following return codes (in hexadecimal form) in register 15:

<table>
<thead>
<tr>
<th>Hex</th>
<th>(Decimal)</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(0)</td>
<td>The exit has completed successfully. Control proceeds to the RACF front-end routine for further security processing and an invocation of RACF.</td>
</tr>
<tr>
<td>C8</td>
<td>(200)</td>
<td>The exit has completed successfully. The SAF router translates this return code to a router return code of 0 and returns control to the issuer of the RACROUTE macro, bypassing RACF processing. (See the note below.)</td>
</tr>
<tr>
<td>CC</td>
<td>(204)</td>
<td>The exit has completed successfully. The SAF router translates this return code to a router return code of 4 and returns control to the issuer of the RACROUTE macro, bypassing RACF processing. (See the note below.)</td>
</tr>
<tr>
<td>D0</td>
<td>(208)</td>
<td>The exit has completed processing. The SAF router translates this return code to a router return code of 8 and returns control to the issuer of the RACROUTE macro, bypassing RACF processing. (See the note below.)</td>
</tr>
<tr>
<td>Other</td>
<td></td>
<td>If the exit routine sets any return code other than those described above, the SAF router returns control directly to the issuer of the RACROUTE macro and passes the untranslated code as the router return code. Further RACF processing is bypassed.</td>
</tr>
</tbody>
</table>
Exit routine processing
Normally, a caller, such as DFP, IMS™, JES, for example, invokes the SAF router and passes it class, requester, and subsystem parameters by way of the RACROUTE exit parameter list. Using those parameters, the SAF router calls the router exit, which then returns to the router with a return code. If the return code is 0, as defined above, the router invokes RACF. RACF reports the results of that invocation to the router by entering return and reason codes in registers 15 and 0, respectively. The router converts the RACF return and reason codes to router return and reason codes and passes them to the caller. The router provides additional information to the caller by placing the unconverted RACF return and reason codes in the first and second words of the router input parameter list.

Installations that use the SAF router exits (ICHRTX00 and ICHRTX01) to modify RACF parameter lists and do customized security processing within the SAF router and that are sensitive to user ID and password parameter values may need to be aware that a user ID that is prefixed by “**”(X'5C5C') indicates an identity context reference. Specifically, RACROUTE REQUEST=VERIFY accepts an identity context reference as an alternative to a user ID and password. Installation exits that examine or manipulate these parameter values will need to determine what to do when an identity context reference, representing a user who has already authenticated, is encountered. While it is possible for an exit to resolve an identity context reference to a security manager (RACF) user ID, it is not recommended, because an identity context reference can only be used one time.

Exit routine requirements
Some RACROUTE requests are service request block (SRB) compatible or cross-memory compatible; that is, they can be issued by an SRB or in cross-memory mode. In addition, some requests are with the local lock held.

Before invoking a system service, a SAF exit must ensure that the service it intends to use is valid for the mode it is running in. For example, many services invoke SVC routines and cannot be used in SRB or cross-memory mode or with a lock held. Another example is cross-memory mode. RACROUTE calls, such as REQUEST=FASTAUTH and REQUEST=TOKENMAP, can be issued in cross-memory mode. A SAF exit cannot issue requests that violate cross-memory mode. One suggestion is that the SAF exit examine the request type in the SAF parameter list and base its processing on that value before it invokes any system service.

Using CICS as an example, CICS passes user installation data in the RACF parameter lists (with INSTLN=, where available) if ESMEXIT=INSTLN was specified as a CICS initialization parameter. The RACF parameter list, pointed to by the SAF parameter list which the exit receives, has a pointer to installation data supplied by the caller (in this case, CICS). The exit can determine that the caller is CICS from the installation data supplied and returned to SAF without taking any further actions.

Some products, such as CICS TS, can issue RACROUTE requests in a performance-sensitive path. If the exit causes a suspension of the caller (for example, a WAIT) then the performance of the caller might suffer. Therefore it is recommended that the exit does not cause suspension during the processing of such performance-sensitive RACROUTE calls.
Simulating a call to RACF

Instead of invoking RACF processing, your installation might choose to have the SAF router exit respond to the caller’s request. If it does, you must still provide the caller with the RACF return and reason codes that it expects to receive. To do so, you must set the router-exit return code, as defined above, so that RACF is not invoked. However, you must still simulate the results of a RACF invocation by coding the exit so it places the RACF return and reason codes in the first and second fullwords, respectively, of the RACROUTE input parameter list.

Note: RACF return and reason codes are documented after the keyword parameter descriptions for the Standard Form of each request type in Chapter 2, “RACF system macros,” on page 7.

JES handling of the return codes from SAF

The default return code of X’04’ from SAF, which is always returned when no SAF router exit is installed (or when return code of X’CC’ is issued from the router exit), causes the job to be queued for execution in the normal manner. That is, JES does user ID and password validation at initiator time.

When the installation has installed a SAF router exit (and within it logically selected the REQUEST=VERIFY, ENVIR=VERIFY call), a successful return code of X’00’ can be returned to JES (by issuing a return code of X’C8’ from the router exit). In this case, the job is queued for execution with the bypass-password-validation indicator set. As a result, password validation, password warning messages, and password-history updates are subsequently bypassed during RACF initialization processing.

The SAF router exit can also cause a failure return code of X’08’ to be returned to JES (by issuing a return code of X’D0’ from the router exit), in which case the job is failed immediately.

Note that if RACF is invoked with the RACINIT macro from the SAF router exit, a RACF abend could cause a JES abend if there is insufficient storage available for JES subtask-recovery termination processing.

SAF callable services router installation exit (IRRSXT00)

The SAF callable services router installation exit IRRSXT00 can be used to add to or replace the functions provided by RACF’s callable services for z/OS UNIX.

IRRSXT00 is called each time a RACF callable service for z/OS UNIX is invoked. The exit is called both before and after RACF is called. When IRRSXT00 is called before RACF, the exit can request that RACF not be called.

For more information about the installation exit IRRSXT00, see z/OS Security Server RACF Callable Services.
Appendix D. SAF interface to an external security product

Security product router

The following is guidance information for programmers implementing a non-RACF external security-manager product.

If RACF is not present in the system, an installation can use an external (non-RACF) security product to provide system-security functions. An external security product supplies a security-product router module to handle security requests from SAF.

When a control point issues the RACROUTE macro to request a security function that is not completely processed by SAF, the SAF router passes control to the security-product router.

SAF completely processes the following:
- REQUEST=TOKENMAP, REQUEST=TOKENBLD, and REQUEST=TOKENXTR.
- REQUEST=VERIFY when building or deleting ACEEs for trusted-system address spaces.
- REQUEST=VERIFYX for trusted-system address spaces.
- REQUEST=VERIFYX and REQUEST=VERIFY when the security product is not installed, or is not active. (In this case, a default UTOKEN is created and the ACEERACF bit is off.)

SAF also partially processes REQUEST=VERIFYX.

For example, SAF:
- Performs user ID propagation for batch jobs to be executed on the local node.
- Performs user ID propagation during spool-reload functions of JES when the security product does not support the RACF 1.9 work-unit identity functions (that is, when RCVTWUID is off).
- Turns a VERIFYX into a VERIFY with ENVIR=CREATE, which it then calls the security-product router module to perform. SAF extracts the UTOKEN from the ACEE and calls the security-product router module to do a VERIFY with ENVIR=DELETE.

In general, SAF processing or partial processing of requests should be transparent to the security product. When SAF completely processes the request, the security product is not called. When SAF does not do any processing for the request, it calls the security-product router to do the processing required for that request type and those parameters. When SAF partially processes the request, it modifies the request type and the parameters when necessary, and calls the security-product router to do the appropriate processing as needed for the resulting request type and parameters.

The external security product must place the address of the security-product router module (which must be in common storage) into field SAFVRACR in the router vector table (mapped by ICHSAFV). The router vector table is built during SAF initialization, but the field SAFVRACR should be filled during the initialization of
Requirements for the security product router

The following general requirements apply for the security-product router module.

**Authorization:**
Supervisor state or problem state, in any PSW key. Generally, this is the state and key of the RACROUTE issuer. However, if SAF is partially processing the request (as explained above), SAF might have switched to supervisor state and key 0 before calling the security-product router.

**Dispatchable unit mode:**
Task mode or SRB mode, as determined by the issuer of RACROUTE.

**Cross-memory mode:**
PASN = HASN = SASN except for:
- RACROUTE REQUEST=TOKENMAP with XMREQ=YES
- RACROUTE REQUEST=TOKENXTR with XMREQ=YES
- RACROUTE REQUEST=FASTAUTH

**Amode:**
24-bit or 31-bit, as determined by the issuer of RACROUTE

**ASC mode:**
Primary

**Locks:** No locks held, except the LOCAL lock can be held when REQUEST=FASTAUTH, and CLASS=PROGRAM, REQSTOR=PADSCHK and SUBSYS=CONTENTS are specified. When called with the LOCAL lock held, the security product must not release the lock.

**Linkage conventions:**
Standard

Input parameters to the security product router

On entry to the security-product router module, register 1 contains the address of the following areas:

<table>
<thead>
<tr>
<th>Offset</th>
<th>Length</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4</td>
<td>Parameter-list address points to the RACROUTE parameter list (mapped by macro ICHSAFP). See data area SAFP in “SAFP: SAF Router Parameter List” in z/OS Security Server RACF Data Areas</td>
</tr>
</tbody>
</table>
SAF interface

The field SAFPRACP in the RACROUTE parameter list contains the offset from the base address of ICHSAFP to the request-specific portion of the parameter list. The format of the request-specific portions varies, depending on the REQUEST= parameter coded on the RACROUTE invocation. The following table shows which request-specific parameter list is assigned to each REQUEST= keyword:

<table>
<thead>
<tr>
<th>RACROUTE REQUEST type</th>
<th>Request-specific parameter list</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQUEST=AUDIT</td>
<td>AUL</td>
</tr>
<tr>
<td>REQUEST=AUTH</td>
<td>ACHKL</td>
</tr>
<tr>
<td>REQUEST=DEFINE</td>
<td>RDDFL</td>
</tr>
<tr>
<td>REQUEST=DIRAUTH</td>
<td>DAUT</td>
</tr>
<tr>
<td>REQUEST=EXTRACT</td>
<td>RXTL</td>
</tr>
<tr>
<td>REQUEST=FASTAUTH</td>
<td>FAST</td>
</tr>
<tr>
<td>REQUEST=LIST</td>
<td>RLST</td>
</tr>
<tr>
<td>REQUEST=SIGNON</td>
<td>SGNPL</td>
</tr>
<tr>
<td>REQUEST=STAT</td>
<td>STAT</td>
</tr>
<tr>
<td>REQUEST=TOKENBLD</td>
<td>RIPL</td>
</tr>
<tr>
<td>REQUEST=TOKENMAP</td>
<td>TSRV</td>
</tr>
<tr>
<td>REQUEST=TOKENXTR</td>
<td>TSRV</td>
</tr>
<tr>
<td>REQUEST=VERIFY</td>
<td>RIPL</td>
</tr>
<tr>
<td>REQUEST=VERIFYX</td>
<td>RIPL</td>
</tr>
</tbody>
</table>

The parameter lists exhibited in the previous tables are illustrated in z/OS Security Server RACF Data Areas. See this text for details on the format of these parameter lists.

Exit conditions from the security product router

On exit from the security-product router module and return to the SAF router:

- Register 15 contains one of the following return codes:

<table>
<thead>
<tr>
<th>Hex (Decimal)</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (0)</td>
<td>The requested function failed.</td>
</tr>
<tr>
<td>4 (4)</td>
<td>The requested function was not processed.</td>
</tr>
<tr>
<td>8 (8)</td>
<td>The requested function was processed, and failed.</td>
</tr>
</tbody>
</table>

- The RACROUTE parameter list (SAFP) contains the function return code in field SAFPRRET and the function reason code in field SAFPRREA for the requested function.
SAF interface

- Register 1 can contain the address of a data area returned by the RACROUTE invocation. See each specific RACROUTE request type for details.

Programming considerations

When an external security product processes a RACROUTE REQUEST=VERIFY,ENVIR=CREATE request, SAF expects the product to build data area ACEE, which is mapped by macro IHAACEE. The ACEE address should be returned as follows:

- If the RACROUTE REQUEST=VERIFY specified the ACEE= keyword, the address of the ACEE should be returned in the word specified by the ACEE= keyword;
- Otherwise, the address should be returned in the TCB field TCBSNV. If the ASXB field ASXBSNV is 0, the address should be returned in ASXBSNV, too. Note that ACEEs whose addresses are in TCBSNV or ASXBSNV must be created in storage located below 16MB.

For token processing, SAF uses field ACEETOKP to reference the token. Token processing is not required but, to be fully compatible with MVS, the security product should create a version 2 ACEE when invoked for RACROUTE REQUEST=VERIFY,ENVIR=CREATE, and should create a token and place its address in ACEETOKP. A security product that is prepared to do this should also set RCVTWUID on to indicate to SAF that it understands token processing and can handle all the token-related fields on a REQUEST=VERIFY call to the security-product router.

In order to be compatible with SAF and various IBM program products, the security product should fill in any field in the ACEE that is listed as a General-Use Programming Interface, with the exception of ACEEFCGP.

In order to be compatible with SAF and various IBM program products, the security product should also process any of the RACROUTE requests and keywords that are listed as General-Use Programming Interfaces, and should provide the same return codes and reason codes as RACF would. If the security product cannot process a specific function, it should return a 4 in register 15, and should set the SAFPRRET and SAFPRREA fields to 0. However, the issuer of the RACROUTE macro is free to interpret return code 4 as desired, and can decide either to accept or to fail the processing that caused the RACROUTE to be issued.

A product can issue some RACROUTE requests in a non-authorized state. The security product must be sensitive to such calls and only invoke services that are allowed in the issuer’s current state. In addition, some RACROUTE requests might be in a performance-sensitive path. If the security product causes a suspension of the caller (for example, a WAIT) then the performance of the caller might suffer. An example of this is CICS, which issues a RACROUTE REQUEST=VERIFY,ENVIR=VERIFY in a non-authorized state and in a performance-sensitive path.

In general, a product that issues RACROUTE should document the minimum level of RACF that is required for the product to work correctly, and any other security product that a customer wishes to use must supply at a minimum the functions that level of RACF would supply. Also, where data is being returned, the data should be in the same format as RACF would return it. The exceptions to this are:

- For RACROUTE REQUEST=EXTRACT,TYPE=EXTRACT or TYPE=REPLACE, the RELEASE=1.8 capabilities of FIELDS= do not have to be supported when the
segment name specified or implied is BASE. Other segment names (such as DFP and TSO) should be supported, however, for compatibility with the RACF releases that support those segments.

In order to be further compatible with SAF and various IBM program products, the security product should fill in any field in the RCVT, which is listed as a General-Use Programming Interface. This includes the fields RCVTREXP and RCVTFRCP, which are used by the RACSTAT and FRACHECK macros. The security product should also assume that references to RACF in the general-use fields apply to other security products as well. For example, RCVTRNA, which is described as “RACF not active”, should be interpreted as “Security product not active”.

Also, although the preferred interface for security services is through RACROUTE, the security product should support the SVC-entered functions RACHECK, RACDEF, RACINIT, RACLST, and RACXTRT. It should also support the branch-entered functions FRACHECK and RACSTAT. It is not necessary to support the ICHEINTY service, which is described in z/OS Security Server RACF Macros and Interfaces.

SAF Callable Services Router Installation exit (IRRSXT00) for z/OS UNIX callable services

The SAF callable services router installation exit IRRSXT00 can be used to add to or replace the functions provided by RACF’s callable services for z/OS UNIX.

IRRSXT00 is called each time a RACF callable service for z/OS UNIX is invoked. The exit is called both before and after RACF is called. When IRRSXT00 is called before RACF, the exit can request that RACF not be called.

An installation implementing z/OS UNIX security support provides its own version of the RACF module that is called by SAF callable services router. The address of this module must be placed into field SAFVXIT2 in the SAF vector table (mapped by ICHSAFV).

For more information about the installation exit IRRSXT00, see z/OS Security Server RACF Callable Services.
Appendix E. Supplied class descriptor table entries

Table 25 lists the class entries supplied by IBM in the class descriptor table (ICHRRCDX). Other classes can be added to the class descriptor table (CDT) by your installation.

If you share a database between MVS and VM systems, you can use the VM classes in the CDT.

**Programming interface information:** The class descriptor table (ICHRRCDX) is mapped by the ICHPCNST macro. Programming interface information for ICHPCNST is found in the data area section called CSNT/CSNX (RACF) in **Security Server RACF Data Areas**

Table 25. Classes supplied by IBM

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCTNUM</td>
<td>POSIT=126 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td>OPER=NO</td>
<td>ID=46</td>
</tr>
<tr>
<td>FIRST=ANY</td>
<td></td>
</tr>
<tr>
<td>ACICSPCT</td>
<td>POSIT=5 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=13</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td>GROUP=BCICSPCT</td>
<td></td>
</tr>
<tr>
<td>OPER=NO</td>
<td>ID=37</td>
</tr>
<tr>
<td>FIRST=ANY</td>
<td></td>
</tr>
<tr>
<td>AIMS</td>
<td>POSIT=4 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td>OPER=NO</td>
<td>ID=11</td>
</tr>
<tr>
<td>FIRST=ALPHA</td>
<td></td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALCSAUTH</td>
<td>POSIT=548 OTHER=ANY RACLIST=ALLOWED MAXLNTH=62</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=YES ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>APPCLU</td>
<td>POSIT=118 OTHER=ANY RACLIST=DISALLOWED MAXLNTH=35</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=57</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>APPCPORT</td>
<td>POSIT=87 OTHER=ANY RACLIST=ALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES RVRSMAC=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=98</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA MAXLENX=17</td>
</tr>
<tr>
<td>APPCSERV</td>
<td>POSIT=84 OTHER=ANY RACLIST=ALLOWED MAXLNTH=73</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=105</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
</tbody>
</table>

CDT entries

398 z/OS V2R1.0 Security Server RACROUTE Macro Reference
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPCSI</td>
<td>POSIT=88 Other=ANY</td>
</tr>
<tr>
<td></td>
<td>RACL=ALLOWED MaxLnth=26</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=READ</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=97</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>APPCTP</td>
<td>POSIT=89 Other=ANY</td>
</tr>
<tr>
<td></td>
<td>RACL=ALLOWED MaxLnth=82</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=96</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>APPL</td>
<td>POSIT=3 Other=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACL=ALLOWED MaxLnth=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>EQUALMAC=YE5</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=8</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>BCICSPCT</td>
<td>POSIT=5 Other=ANY</td>
</tr>
<tr>
<td></td>
<td>MaxLnth=13 DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=ACICSPCT</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=38</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>

---

**Appendix E. Supplied class descriptor table entries**
### CDT entries

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CACHECLS</td>
<td>POSIT=569 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=16</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=NO</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=YES ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>CBIND</td>
<td>POSIT=545 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=41</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>CCICSCMD</td>
<td>POSIT=5 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=21</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=VCICSCMD</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=52</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>CDT</td>
<td>POSIT=572 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=NO</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=YES ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>SIGNAL=YES GENERIC=DISALLOWED</td>
</tr>
<tr>
<td>Class</td>
<td>Attributes</td>
</tr>
<tr>
<td>----------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>CFLD</td>
<td>POSIT=588 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=DISALLOWED MAXLNTH=26</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED</td>
</tr>
<tr>
<td></td>
<td>RAQLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=1 FIRST=ALPHA SIGNAL=NO GENERIC=DISALLOWED</td>
</tr>
<tr>
<td>CIMS</td>
<td>POSIT=93 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLST=DISALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE GROUP=DIMS</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=88 FIRST=ALPHA</td>
</tr>
<tr>
<td>CONSOLE</td>
<td>POSIT=107 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=ALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE RVRSMAC=YES OPER=NO ID=68 FIRST=ANY</td>
</tr>
<tr>
<td>CPSMOBJ</td>
<td>POSIT=57 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=ALLOWED MAXLNTH=44</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>GROUP=GCPSMOBJ OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1 FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPSMXMP</td>
<td>POSIT=11 OTHER=ANY RACLIST=ALLOWED MAXLNTH=44</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4 OPER=NO ID=1 FIRST=ANY</td>
</tr>
<tr>
<td>CRYPTOZ</td>
<td>POSIT=578 OTHER=ANY RACLIST=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4 RACLREQ=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO KEYQUAL=0 ID=1 FIRST=ANY</td>
</tr>
<tr>
<td>CSFKEYS</td>
<td>POSIT=98 OTHER=ANY RACLIST=ALLOWED MAXLNTH=73</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4 RACLREQ=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GCSFKEYS OPER=NO ID=100 FIRST=ALPHA</td>
</tr>
<tr>
<td>CSFSERV</td>
<td>POSIT=98 OTHER=ALPHANUM RACLIST=ALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4 RACLREQ=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=99 FIRST=ALPHA</td>
</tr>
</tbody>
</table>
## CDT entries

**Table 25. Classes supplied by IBM (continued)**

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>DASDVOL</td>
<td>POSIT=0, OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED, MAXLNTH=6</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>GROUP=GADSDVOL</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=5</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>DBNFORM</td>
<td>POSIT=59, OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED, MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4, DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>DCEUUDS</td>
<td>POSIT=544, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED, MAXLNTH=73</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED, DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>DCICSDCT</td>
<td>POSIT=5, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=13, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=ECICSDCT</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=31</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>

Appendix E. Supplied class descriptor table entries  403
<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEVICES</td>
<td>POSIT=115 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACL=ALLOWED MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAACLREQ=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=60 FIRST=ANY</td>
</tr>
<tr>
<td>DIGTCERT</td>
<td>POSIT=550 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACL=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAACLREQ=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO KEYQUAL=0 ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>DIGTCRT</td>
<td>POSIT=563 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACL=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAACLREQ=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO KEYQUAL=0 ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>DIGTNMAP</td>
<td>POSIT=563 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACL=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAACLREQ=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO KEYQUAL=0 ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>Class</td>
<td>Attributes</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>DIGTRING</td>
<td>POSIT=550, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACL=ALLOWED, MAXL=246</td>
</tr>
<tr>
<td></td>
<td>GENL=DISALLOWED, DFTRET=4</td>
</tr>
<tr>
<td></td>
<td>RAQLRE=NO, DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, KEYQUAL=0</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td>DIMS</td>
<td>POSIT=93, OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACL=ALLOWED, MAXL=8</td>
</tr>
<tr>
<td></td>
<td>GENL=DISALLOWED, DFTRET=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=CIMS</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=89</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>DIRACC</td>
<td>POSIT=71, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACL=DISALLOWED, MAXL=246</td>
</tr>
<tr>
<td></td>
<td>GENL=DISALLOWED, DFTRET=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=NO, ID=107</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>DIRAUTH</td>
<td>POSIT=105, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACL=DISALLOWED, MAXL=8</td>
</tr>
<tr>
<td></td>
<td>GENL=DISALLOWED, DFTRET=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=NO, ID=70</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIRECTRY</td>
<td>POSIT=95, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED, MAXLNTH=153</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=YES, KEYQUAL=2</td>
</tr>
<tr>
<td></td>
<td>ID=86</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>DIRSRCH</td>
<td>POSIT=70, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED, MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=NO, ID=106</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>DLFCLASS</td>
<td>POSIT=92, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED, MAXLNTH=64</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=90</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>DSNADM</td>
<td>POSIT=539, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED, MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLI REQ=NO, DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=YES, ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td></td>
<td>SIGNAL=YES, MAXLENX=246</td>
</tr>
</tbody>
</table>
### CDT entries

**Table 25. Classes supplied by IBM (continued)**

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSNR</td>
<td>POSIT=7, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED, MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES, EQUALMAC=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=18</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>ECICSDCT</td>
<td>POSIT=5, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=13, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=DCICSDCT</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=32</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>EJBROLE</td>
<td>POSIT=568, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED, MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO, DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GEJBROLE, SLBLREQ=NO</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=1</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=YES, CASE=ASIS</td>
</tr>
<tr>
<td>FACILITY</td>
<td>POSIT=8, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED, MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=19</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
</table>
| FCICSFCT | POSIT=5 OTHER=ANY  
RACLIST=ALLOWED MAXLNGTH=17  
DFTRETC=4  
DFTUACC=NONE  
GROUP=HCICSFCT  
OPER=NO  
ID=27  
FIRST=ANY |
| FIELD | POSIT=121 OTHER=ANY  
RACLIST=ALLOWED MAXLNGTH=26  
GENLIST=ALLOWED DFTRETC=4  
DFTUACC=NONE  
RACLREQ=YES  
OPER=NO  
ID=51  
FIRST=ALPHA |
| FILE | POSIT=94 OTHER=ANY  
RACLIST=DISALLOWED MAXLNTH=171  
GENLIST=DISALLOWED DFTRETC=8  
DFTUACC=NONE  
SLBLREQ=YES  
OPER=YES KEYQUAL=2  
ID=87  
FIRST=ANY |
| FIMS | POSIT=101 OTHER=ALPHANUM  
RACLIST=DISALLOWED MAXLNTH=8  
GENLIST=DISALLOWED DFTRETC=4  
DFTUACC=NONE  
GROUP=HIMS  
OPER=NO  
ID=79  
FIRST=ALPHANUM |
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSACCESS</td>
<td>POSIT=595, OTHER=ANY, RACLIST=ALLOWED, MAXLNTH=44, GENLIST=DISALLOWED,</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4, RACLREQ=YES, DFTUACC=NONE, SLBLREQ=NO, OPER=NO, KEYQUAL=0,</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=YES, ID=1, FIRST=ANY, CASE=UPPER, SIGNAL=YES, GENERIC=ALLOWED</td>
</tr>
<tr>
<td>FSOBJ</td>
<td>POSIT=72, OTHER=ANY, RACLIST=DISALLOWED, MAXLNTH=246, GENLIST=DISALLOWED</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8, DFTUACC=NONE, OPER=NO, PROFDEF=NO, ID=108, FIRST=ANY</td>
</tr>
<tr>
<td>FSSEC</td>
<td>POSIT=73, OTHER=ANY, RACLIST=DISALLOWED, MAXLNTH=246, GENLIST=DISALLOWED</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8, DFTUACC=NONE, OPER=NO, PROFDEF=NO, ID=109, FIRST=ANY</td>
</tr>
<tr>
<td>GCICSTRN</td>
<td>POSIT=5, OTHER=ANY, MAXLNTH=13, DFTRETC=4, DFTUACC=NONE, MEMBER=TCICSTRN</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=13, FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCPSMOBJ</td>
<td>POSIT=57, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=246, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>GCSFKEYS</td>
<td>POSIT=98, OTHER=NONATNUM</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED</td>
</tr>
<tr>
<td></td>
<td>RAACLREQ=NO, DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>ID=101</td>
</tr>
<tr>
<td></td>
<td>FIRST=NONATABC</td>
</tr>
<tr>
<td>GDASDVOL</td>
<td>POSIT=0, OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=6, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>ID=39</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>GDSNBP</td>
<td>POSIT=536, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED</td>
</tr>
<tr>
<td></td>
<td>RAACLREQ=NO, DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES, PROFDEF=YES</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>
### CDT entries

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GDSNCL</td>
<td>POSIT=538, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RAONLY=DISALLOWED, MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAONLY=NO, DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MDSNCL</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=YES, ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLENX=246</td>
</tr>
<tr>
<td>GDSNDB</td>
<td>POSIT=528, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RAONLY=DISALLOWED, MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAONLY=NO, DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MDSNDB</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=YES, ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>GDSNGV</td>
<td>POSIT=596, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RAONLY=DISALLOWED, MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAONLY=NO, DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MDSNGV</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=YES, ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLENX=246</td>
</tr>
<tr>
<td></td>
<td>SIGNAL=YES</td>
</tr>
</tbody>
</table>

---

**Table 25. Classes supplied by IBM (continued)**

Appendix E. Supplied class descriptor table entries 411
<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GDSNJR</td>
<td>POSIT=567 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAQLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MDSNJR OPER=NO PROFDEF=YES ID=1 FIRST=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLENX=246</td>
</tr>
<tr>
<td>GDSNPK</td>
<td>POSIT=534 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAQLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=NO</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MDSNPK OPER=NO PROFDEF=YES ID=1 FIRST=ANY SIGNAL=YES</td>
</tr>
<tr>
<td>GDSNPN</td>
<td>POSIT=533 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAQLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=Yes</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MDSNPN OPER=NO PROFDEF=YES ID=1 FIRST=ANY</td>
</tr>
<tr>
<td>GDSNSC</td>
<td>POSIT=562 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAQLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=Yes</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MDSNSC OPER=NO PROFDEF=YES ID=1 FIRST=ANY</td>
</tr>
</tbody>
</table>

MAXLENX=246
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GDSNSG</td>
<td>POSIT=537 OTHER=ANY RACLIST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4 RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MDSNSG OPER=NO PROFDEF=YES ID=1 FIRST=ANY</td>
</tr>
<tr>
<td></td>
<td>SIGNAL=YES MAXLENX=246</td>
</tr>
<tr>
<td>GDSNSM</td>
<td>POSIT=535 OTHER=ANY RACLIST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4 RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MDSNSM OPER=NO PROFDEF=YES ID=1 FIRST=ANY</td>
</tr>
<tr>
<td></td>
<td>SIGNAL=YES</td>
</tr>
<tr>
<td>GDSNSP</td>
<td>POSIT=561 OTHER=ANY RACLIST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4 RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MDSNSP OPER=NO PROFDEF=YES ID=1 FIRST=ANY</td>
</tr>
<tr>
<td></td>
<td>SIGNAL=YES MAXLENX=246</td>
</tr>
<tr>
<td>GDSNSQ</td>
<td>POSIT=573 OTHER=ANY RACLIST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4 RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MDSNSQ OPER=NO PROFDEF=YES ID=1 FIRST=ANY</td>
</tr>
<tr>
<td></td>
<td>SIGNAL=YES MAXLENX=246</td>
</tr>
<tr>
<td>Class</td>
<td>Attributes</td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>GDSNTB</td>
<td>POSIT=530, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO, MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=NO, MEMBER=MDSNTB</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, PROFDEF=YES, ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY, SIGNAL=YES, MAXLENX=246</td>
</tr>
<tr>
<td>GDSNTS</td>
<td>POSIT=529, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO, MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=NO, MEMBER=MDSNTS</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, PROFDEF=YES, ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY, SIGNAL=YES, MAXLENX=246</td>
</tr>
<tr>
<td>GDSNUF</td>
<td>POSIT=560, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO, MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=NO, MEMBER=MDSNUF</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, PROFDEF=YES, ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY, SIGNAL=YES, MAXLENX=246</td>
</tr>
<tr>
<td>GDSNUT</td>
<td>POSIT=559, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO, MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=NO, MEMBER=MDSNUT</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, PROFDEF=YES, ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY, MAXLENX=246</td>
</tr>
</tbody>
</table>
### CDT entries

Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GEJBROLE</td>
<td>POSIT=568 OTHER=ANY RACLST=DISALLOWED MAXLNTH=246 GENLIST=DISALLOWED DFTRETC=4 RACLREQ=NO DFTUACC=NONE SLBLREQ=NO MEMBER=GEJBROLE OPER=NO PROFDEF=YES ID=1 FIRST=ANY CASE=ASIS</td>
</tr>
<tr>
<td>GIMS</td>
<td>POSIT=4 OTHER=ALPHANUM MAXLNTH=8 DFTRETC=4 DFTUACC=NONE MEMBER=TIMS OPER=NO ID=10 FIRST=ALPHA</td>
</tr>
<tr>
<td>GINFOMAN</td>
<td>POSIT=85 OTHER=ANY RACLST=DISALLOWED MAXLNTH=44 GENLIST=DISALLOWED DFTRETC=4 MEMBER=GINFOMAN OPER=NO ID=104 FIRST=ANY</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>POSIT=6 OTHER=ANY MAXLNTH=8 DFTRETC=4 DFTUACC=NONE MEMBER=GMBR OPER=NO ID=17 FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMBR</td>
<td>POSIT=6 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GLOBAL</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=16</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>GMQADMIN</td>
<td>POSIT=80 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=62</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MQADMIN</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=121</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>GMQCHAN</td>
<td>POSIT=58 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MQCHAN</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>GMQNLIST</td>
<td>POSIT=79 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MQNLIST</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=119</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>
### CDT entries

**Table 25. Classes supplied by IBM (continued)**

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMQPROC</td>
<td>POSIT=78 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MQPROC OPER=NO ID=117 FIRST=ANY</td>
</tr>
<tr>
<td>GMQQUEUE</td>
<td>POSIT=77 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MQQUEUE OPER=NO ID=4 FIRST=ANY</td>
</tr>
<tr>
<td>GMXADMIN</td>
<td>POSIT=586 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=62</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MXADMIN OPER=NO ID=1 FIRST=ANY CASE=ASIS</td>
</tr>
<tr>
<td>GMXNLIST</td>
<td>POSIT=585 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MXNLIST OPER=NO ID=1 FIRST=ANY CASE=ASIS</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMXPROC</td>
<td>POSIT=584 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MXPROC OPER=NO ID=1 FIRST=ANY CASE=ASIS</td>
</tr>
<tr>
<td>GMXQUEUE</td>
<td>POSIT=583 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MXQUEUE OPER=NO ID=1 FIRST=ANY CASE=ASIS</td>
</tr>
<tr>
<td>GMXTOPIC</td>
<td>POSIT=587 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MXTOPIC OPER=NO ID=1 FIRST=ANY CASE=ASIS</td>
</tr>
<tr>
<td>GS NSF</td>
<td>POSIT=100 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=63</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=SDSF OPER=NO ID=95 FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSOMDOBJ</td>
<td>(\text{POSIT}=547) (\text{OTHER}=\text{ANY})</td>
</tr>
<tr>
<td></td>
<td>(\text{MAXLNTH}=246)</td>
</tr>
<tr>
<td></td>
<td>(\text{DFTRETC}=8)</td>
</tr>
<tr>
<td></td>
<td>(\text{MEMBER}=\text{SOMDOBJ})</td>
</tr>
<tr>
<td></td>
<td>(\text{OPER}=\text{NO})</td>
</tr>
<tr>
<td></td>
<td>(\text{ID}=1)</td>
</tr>
<tr>
<td></td>
<td>(\text{FIRST}=\text{ALPHA})</td>
</tr>
<tr>
<td>GTERMINL</td>
<td>(\text{POSIT}=2) (\text{OTHER}=\text{ALPHANUM})</td>
</tr>
<tr>
<td></td>
<td>(\text{MAXLNTH}=8)</td>
</tr>
<tr>
<td></td>
<td>(\text{DFTRETC}=4)</td>
</tr>
<tr>
<td></td>
<td>(\text{MEMBER}=\text{TERMINAL})</td>
</tr>
<tr>
<td></td>
<td>(\text{OPER}=\text{NO})</td>
</tr>
<tr>
<td></td>
<td>(\text{ID}=42)</td>
</tr>
<tr>
<td></td>
<td>(\text{FIRST}=\text{ALPHANUM})</td>
</tr>
<tr>
<td>GXCSFKEY</td>
<td>(\text{POSIT}=98) (\text{OTHER}=\text{ALPHANUM})</td>
</tr>
<tr>
<td></td>
<td>(\text{RACLISTER}=\text{DISALLOWED}) (\text{MAXLNTH}=246)</td>
</tr>
<tr>
<td></td>
<td>(\text{GENLISTER}=\text{DISALLOWED})</td>
</tr>
<tr>
<td></td>
<td>(\text{DFTUACC}=\text{NONE})</td>
</tr>
<tr>
<td></td>
<td>(\text{MEMBER}=\text{XCSFKEY})</td>
</tr>
<tr>
<td></td>
<td>(\text{OPER}=\text{NO})</td>
</tr>
<tr>
<td></td>
<td>(\text{ID}=101)</td>
</tr>
<tr>
<td></td>
<td>(\text{FIRST}=\text{ALPHA})</td>
</tr>
<tr>
<td>GXFACILI</td>
<td>(\text{POSIT}=8) (\text{OTHER}=\text{ANY})</td>
</tr>
<tr>
<td></td>
<td>(\text{RACLISTER}=\text{ALLOWED}) (\text{MAXLNTH}=246)</td>
</tr>
<tr>
<td></td>
<td>(\text{GENLISTER}=\text{ALLOWED}) (\text{DFTRETC}=4)</td>
</tr>
<tr>
<td></td>
<td>(\text{DFTUACC}=\text{NONE})</td>
</tr>
<tr>
<td></td>
<td>(\text{MEMBER}=\text{XFACILIT})</td>
</tr>
<tr>
<td></td>
<td>(\text{OPER}=\text{NO})</td>
</tr>
<tr>
<td></td>
<td>(\text{ID}=1)</td>
</tr>
<tr>
<td></td>
<td>(\text{FIRST}=\text{ALPHANUM})</td>
</tr>
</tbody>
</table>
### CDT entries

**Table 25. Classes supplied by IBM (continued)**

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GZMFAPLA</td>
<td>POSIT=592 OTHER=ANY RACLIST=DISALLOWED MAXLNTH=246 GENLIST=DISALLOWED DFTRETC=4 RAACLREQ=NO DFTUACC=NONE SLBLREQ=NO MEMBER=ZMFAPLA OPER=NO PROFDEF=YES ID=1 FIRST=ANY CASE=ASIS</td>
</tr>
<tr>
<td>HCICSFCT</td>
<td>POSIT=5 OTHER=ANY MAXLNTH=17 DFTRETC=4 DFTUACC=NONE MEMBER=FCICSFCT OPER=NO ID=28 FIRST=ANY</td>
</tr>
<tr>
<td>HIMS</td>
<td>POSIT=101 OTHER=ALPHANUM RACLIST=DISALLOWED MAXLNTH=8 GENLIST=DISALLOWED DFTRETC=4 DFTUACC=NONE MEMBER=FIMS OPER=NO ID=80 FIRST=ALPHANUM</td>
</tr>
<tr>
<td>IBMOPC</td>
<td>POSIT=60 OTHER=ANY MAXLNTH=60 DFTRETC=4 DFTUACC=NONE OPER=NO ID=1 FIRST=ALPHA</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDIDMAP</td>
<td>ID=1&lt;br&gt;POSIT=591&lt;br&gt;RACLIST=ALLOWED&lt;br&gt;MAXLNTH=246&lt;br&gt;GENLIST=DISALLOWED&lt;br&gt;RACLREQ=YES&lt;br&gt;DFTUACC=NONE&lt;br&gt;OPER=NO&lt;br&gt;KEYQUAL=0&lt;br&gt;ID=1&lt;br&gt;FIRST=ANY&lt;br&gt;GENERIC=DISALLOWED</td>
</tr>
<tr>
<td>IIMS</td>
<td>ID=1&lt;br&gt;POSIT=575&lt;br&gt;RACLIST=DISALLOWED&lt;br&gt;MAXLNTH=8&lt;br&gt;GENLIST=DISALLOWED&lt;br&gt;DFTUACC=NONE&lt;br&gt;GROUP=JIMS&lt;br&gt;OPER=NO&lt;br&gt;ID=1&lt;br&gt;FIRST=ALPHA</td>
</tr>
<tr>
<td>ILMADMIN</td>
<td>ID=1&lt;br&gt;POSIT=566&lt;br&gt;RACLIST=ALLOWED&lt;br&gt;MAXLNTH=246&lt;br&gt;GENLIST=ALLOWED&lt;br&gt;DFTUACC=NONE&lt;br&gt;SLBLREQ=NO&lt;br&gt;OPER=NO&lt;br&gt;ID=1&lt;br&gt;FIRST=ANY</td>
</tr>
<tr>
<td>INFOMAN</td>
<td>ID=103&lt;br&gt;POSIT=85&lt;br&gt;RACLIST=ALLOWED&lt;br&gt;MAXLNTH=44&lt;br&gt;GENLIST=ALLOWED&lt;br&gt;DFTRETC=4&lt;br&gt;GROUP=GINFOMAN&lt;br&gt;OPER=NO&lt;br&gt;ID=103&lt;br&gt;FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPCOBJ</td>
<td>POSIT=62 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=NO; ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>JAVA</td>
<td>POSIT=556 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>JCICSJCT</td>
<td>POSIT=5 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=17</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=KCICSJCT</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=29</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>JESINPUT</td>
<td>POSIT=108 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>EQUALMAC=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=67</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>
### Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>JESJOBS</td>
<td>POSIT=109  OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RA CLIST=ALLOWED MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>GEN LIST=ALLOWED DFTRET=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=66</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>JESSPOOL</td>
<td>POSIT=110  OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RA CLIST=ALLOWED MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>GEN LIST=DISALLOWED DFTRET=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=65</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>JIMS</td>
<td>POSIT=575  OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RA CLIST=DISALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GEN LIST=DISALLOWED DFTRET=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=IIMS</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>KCICSJCT</td>
<td>POSIT=5  OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=17</td>
</tr>
<tr>
<td></td>
<td>DFTRET=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=KCICSJCT</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=30</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>KERBLINK</td>
<td>POSIT=565 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=240</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=None</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>CASE=ASIS ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY GENERIC=DISALLOWED</td>
</tr>
<tr>
<td>KEYSMSTR</td>
<td>POSIT=543 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=None</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>LDAP</td>
<td>POSIT=593 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLIST=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=NO</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY CASE=UPPER</td>
</tr>
<tr>
<td>LDAPBIND</td>
<td>POSIT=571 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>LFSCLASS</td>
<td>POSIT=12 OTHER=ANY RACLIST=ALLOWED MAXLNTH=246 DFTRETC=4 DFTUACC=NONE OPER=NO ID=1 FIRST=ALPHANUM</td>
</tr>
<tr>
<td>LIMS</td>
<td>POSIT=576 OTHER=ALPHANUM RACLIST=DISALLOWED MAXLNTH=8 GENLIST=DISALLOWED DFTRETC=4 DFTUACC=NONE GROUP=MIMS OPER=NO ID=1 FIRST=ALPHANUM</td>
</tr>
<tr>
<td>LOGSTRM</td>
<td>POSIT=61 OTHER=ANY RACLIST=ALLOWED MAXLNTH=26 GENLIST=ALLOWED DFTRETC=4 DFTUACC=NONE OPER=NO ID=1 FIRST=ALPHANUM</td>
</tr>
<tr>
<td>MCICSPPT</td>
<td>POSIT=5 OTHER=ANY MAXLNTH=17 DFTRETC=4 DFTUACC=NONE GROUP=NCICSPPT OPER=NO ID=35 FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDSNBP</td>
<td>POSIT=536 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GDSNBP SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=1</td>
</tr>
<tr>
<td>MDSNCL</td>
<td>POSIT=538 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GDSNCL SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=1</td>
</tr>
<tr>
<td>MDSNDB</td>
<td>POSIT=528 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GDSNDB SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=1</td>
</tr>
<tr>
<td>MDSNGV</td>
<td>POSIT=596 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=DISALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GDSNGV SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY MAXLENX=246</td>
</tr>
<tr>
<td></td>
<td>SIGNAL=YES</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDSNJR</td>
<td>POSIT=567 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GDSNJR SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLENX=246</td>
</tr>
<tr>
<td>MDSNPK</td>
<td>POSIT=534 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GDSNPK SLBLREQ=NO</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY SIGNAL=YES</td>
</tr>
<tr>
<td>MDSNPN</td>
<td>POSIT=533 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GDSNPN SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>MDSNSC</td>
<td>POSIT=562 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GDSNSC SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLENX=246</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDSNSG</td>
<td>POSIT=537 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GDSNSG SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY MAXLENX=246</td>
</tr>
<tr>
<td>MDSNSM</td>
<td>POSIT=535 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GDSNSM SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY SIGNAL=ANY</td>
</tr>
<tr>
<td>MDSNSP</td>
<td>POSIT=561 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GDSNSP SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY SIGNAL=ANY</td>
</tr>
<tr>
<td>MDSNSQ</td>
<td>POSIT=573 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=100</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GDSNSQ SLBLREQ=NO</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY SIGNAL=ANY</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CDT entries
### CDT entries

#### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDSNTB</td>
<td>( \text{POSIT}=530 ), ( \text{OTHER}=\text{ANY} )</td>
</tr>
<tr>
<td></td>
<td>( \text{RACLIST} = \text{DISALLOWED} ), ( \text{MAXLNTH}=100 )</td>
</tr>
<tr>
<td></td>
<td>( \text{GENLIST} = \text{DISALLOWED} ), ( \text{DFTRETC}=4 )</td>
</tr>
<tr>
<td></td>
<td>( \text{RACLREQ}=\text{NO} ), ( \text{DFTUACC} = \text{NONE} )</td>
</tr>
<tr>
<td></td>
<td>( \text{GROUP}=\text{GDSNTB} ), ( \text{SLBLREQ}=\text{YES} )</td>
</tr>
<tr>
<td></td>
<td>( \text{OPER} = \text{NO} )</td>
</tr>
<tr>
<td></td>
<td>( \text{PROFDEF} = \text{YES} ), ( \text{ID}=1 )</td>
</tr>
<tr>
<td></td>
<td>( \text{FIRST} = \text{ANY} )</td>
</tr>
<tr>
<td></td>
<td>( \text{SIGNAL} = \text{YES} ), ( \text{MAXLENX}=246 )</td>
</tr>
</tbody>
</table>

| MDSNTS | \( \text{POSIT}=529 \), \( \text{OTHER}=\text{ANY} \) |
|        | \( \text{RACLIST} = \text{DISALLOWED} \), \( \text{MAXLNTH}=100 \) |
|        | \( \text{GENLIST} = \text{DISALLOWED} \), \( \text{DFTRETC}=4 \) |
|        | \( \text{RACLREQ}=\text{NO} \), \( \text{DFTUACC} = \text{NONE} \) |
|        | \( \text{GROUP}=\text{GDSNTS} \), \( \text{SLBLREQ}=\text{YES} \) |
|        | \( \text{OPER} = \text{NO} \) |
|        | \( \text{PROFDEF} = \text{YES} \), \( \text{ID}=1 \) |
|        | \( \text{FIRST} = \text{ANY} \) |

| MDSNUF | \( \text{POSIT}=560 \), \( \text{OTHER}=\text{ANY} \) |
|        | \( \text{RACLIST} = \text{DISALLOWED} \), \( \text{MAXLNTH}=100 \) |
|        | \( \text{GENLIST} = \text{DISALLOWED} \), \( \text{DFTRETC}=4 \) |
|        | \( \text{RACLREQ}=\text{NO} \), \( \text{DFTUACC} = \text{NONE} \) |
|        | \( \text{GROUP}=\text{GDSNUF} \), \( \text{SLBLREQ}=\text{YES} \) |
|        | \( \text{OPER} = \text{NO} \) |
|        | \( \text{PROFDEF} = \text{YES} \), \( \text{ID}=1 \) |
|        | \( \text{FIRST} = \text{ANY} \) |
|        | \( \text{SIGNAL} = \text{YES} \), \( \text{MAXLENX}=246 \) |

<p>| MDSNUT | ( \text{POSIT}=559 ), ( \text{OTHER}=\text{ANY} ) |
|        | ( \text{RACLIST} = \text{DISALLOWED} ), ( \text{MAXLNTH}=100 ) |
|        | ( \text{GENLIST} = \text{DISALLOWED} ), ( \text{DFTRETC}=4 ) |
|        | ( \text{RACLREQ}=\text{NO} ), ( \text{DFTUACC} = \text{NONE} ) |
|        | ( \text{GROUP}=\text{GDSNUT} ), ( \text{SLBLREQ}=\text{NO} ) |
|        | ( \text{OPER} = \text{NO} ) |
|        | ( \text{PROFDEF} = \text{YES} ), ( \text{ID}=1 ) |
|        | ( \text{FIRST} = \text{ANY} ) |
|        | ( \text{MAXLENX}=246 ) |</p>
<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MGMTCLAS</td>
<td>POSIT=123 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=49</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>MIMS</td>
<td>POSIT=576 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=LIMS OPER=NO ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>MQADMIN</td>
<td>POSIT=80 OTHER=ANY MAXLNTH=62</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8 DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GMQADMIN OPER=NO ID=120</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>MQCHAN</td>
<td>POSIT=58 OTHER=ANY MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8 DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GMQCHAN OPER=NO ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>Class</td>
<td>Attributes</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>MQCMDs</td>
<td>POSIT=81 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=22</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=122</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>MQCONN</td>
<td>POSIT=82 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=10</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>EQUALMAC=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=123</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>MQNLIST</td>
<td>POSIT=79 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GMQNLIST</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=118</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>MQPROC</td>
<td>POSIT=78 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GMQPROC</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=116</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>

Table 25. Classes supplied by IBM (continued)
Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MQQUEUE</td>
<td>POSIT=77 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GMQQUEUE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=2</td>
</tr>
<tr>
<td>MXADMIN</td>
<td>POSIT=586 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=62</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GMXADMIN</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td>MXNLIST</td>
<td>POSIT=585 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GMXNLIST</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td>MXPROC</td>
<td>POSIT=584 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GMXPROC</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY CASE=ASIS</td>
</tr>
</tbody>
</table>
### CDT entries

**Table 25. Classes supplied by IBM (continued)**

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MXQUEUE</td>
<td>POSIT=583 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=53</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GMXQUEUE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=1 FIRST=ANY</td>
</tr>
<tr>
<td></td>
<td>CASE=ASIS</td>
</tr>
<tr>
<td>MXTOPIC</td>
<td>POSIT=587 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GMXTOPIC</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY CASE=ASIS</td>
</tr>
<tr>
<td>NCICSPPT</td>
<td>POSIT=5 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=17</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=MCICSPPT</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=36</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>NDSLINK</td>
<td>POSIT=554 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4 ID=1 FIRST=ANY</td>
</tr>
</tbody>
</table>
### CDT entries

**Table 25. Classes supplied by IBM (continued)**

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>NETCMDS</td>
<td>POSIT=68, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED, MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>NETSPAN</td>
<td>POSIT=67, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>NODES</td>
<td>POSIT=103, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED, GENLIST=DISALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAERQ=YES, GENLIST=DISALLOWED, DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=NODMBR</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=72</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>NODMBR</td>
<td>POSIT=103, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED, MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=NODES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=43</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>
Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOTELINK</td>
<td>POSIT=553 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACTION=DISALLOWED MAXLNTH=64</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=YES DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>NVASAPDT</td>
<td>POSIT=97 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACTION=DISALLOWED MAXLNTH=17</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=83</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>OIMS</td>
<td>POSIT=101 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACTION=DISALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=WIMS</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=81</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>OPERCMDS</td>
<td>POSIT=112 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACTION=ALLOWED MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACTION=ALLOWED DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=63</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCICSPSB</td>
<td>POSIT=5 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=17</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=QCICSPSB</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=14</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>PERFGRP</td>
<td>POSIT=125 OTHER=NUMERIC</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=3</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=47</td>
</tr>
<tr>
<td></td>
<td>FIRST=NUMERIC</td>
</tr>
<tr>
<td>PIMS</td>
<td>POSIT=101 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=QIMS</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=75</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>PKISERV</td>
<td>POSIT=597 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO KEYQUAL=0</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY CASE=UPPER</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMBR</td>
<td>POSIT=13 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=44</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=PROGRAM</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=40</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>PRINTSRV</td>
<td>POSIT=570 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=17</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY MAXLENX=64</td>
</tr>
<tr>
<td>PROCACT</td>
<td>POSIT=75 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=NO ID=111</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>PROCESS</td>
<td>POSIT=74 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO PROFDEF=NO ID=110</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROGRAM</td>
<td>POSIT=13, OTHER=ALPHANUM, MAXLNTH=8, DFTRETC=4, DFTUACC=NONE, MEMBER=PMBR, OPER=NO, ID=41, FIRST=ALPHA</td>
</tr>
<tr>
<td>PROPCNTL</td>
<td>POSIT=119, OTHER=ALPHANUM, MAXLNTH=8, RACLIST=ALLOWED, GENLIST=DISALLOWED, RACLREQ=YES, DFTRETC=4, DFTUACC=NONE, OPER=NO, ID=56, FIRST=ALPHA</td>
</tr>
<tr>
<td>PSFMPL</td>
<td>POSIT=113, OTHER=ANY, RACLIST=ALLOWED, GENLIST=DISALLOWED, RACLREQ=YES, DFTRETC=8, DFTUACC=NONE, OPER=YES, ID=62, FIRST=ANY</td>
</tr>
<tr>
<td>PTKDATA</td>
<td>POSIT=76, OTHER=ANY, RACLIST=ALLOWED, MAXLNTH=39, DFTRETC=4, RACLREQ=YES, DFTUACC=NONE, OPER=NO, ID=112, FIRST=ALPHANUM</td>
</tr>
<tr>
<td>Class</td>
<td>Attributes</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td>PTKTVAL</td>
<td>POSIT=76 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>QCICSPSB</td>
<td>POSIT=5 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=17</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=PCICSPSB</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=15</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>QIMS</td>
<td>POSIT=101 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=PIMS</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=76</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>RACFEVNT</td>
<td>POSIT=574 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>Class</td>
<td>Attributes</td>
</tr>
<tr>
<td>------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>RACFHC</td>
<td>POSIT=590   OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RAclist=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>RAclrreq=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=RACHCMBR</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>RACFVARS</td>
<td>POSIT=102   OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RAclist=ALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=RVARSMBR</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=74</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>RACGLIST</td>
<td>POSIT=10    OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RAclist=DISALLOWED MAXLNTH=14</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAclrreq=NO   DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=NO</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>RACHCMBR</td>
<td>POSIT=590   OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RAclist=DISALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=RACFHC</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAUDITX</td>
<td>POSIT=577 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RAclist=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAclREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>ID=1 FIRST=ANY</td>
</tr>
<tr>
<td>RCICSRES</td>
<td>POSIT=5 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=54</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=WCICSRES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=52 FIRST=ANY CASE=ASIS</td>
</tr>
<tr>
<td>RDATALIB</td>
<td>POSIT=581 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RAclist=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAclREQ=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO KEYQUAL=0</td>
</tr>
<tr>
<td></td>
<td>ID=1 FIRST=ANY CASE=UPPER</td>
</tr>
<tr>
<td>REALM</td>
<td>POSIT=564 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=240</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAclist=ALLOWED</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1 FIRST=ANY GENERIC=DISALLOWED</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>RIMS</td>
<td>POSIT=589, OTHER=ALPHANUM, MAXLNTH=8, GENLIST=DISALLOWED, DFTRETC=4, DFTUACC=NONE, RACLST=DISALLOWED, OPER=NO, ID=1, FIRST=ALPHANUM, CASE=ASIS</td>
</tr>
<tr>
<td>RMTOPS</td>
<td>POSIT=86, OTHER=ANY, MAXLNTH=246, DFTRETC=4, DFTUACC=NONE, OPER=NO, ID=102, FIRST=ALPHA</td>
</tr>
<tr>
<td>RODMGR</td>
<td>POSIT=69, OTHER=ANY, MAXLNTH=8, DFTRETC=4, ID=113, FIRST=ALPHANUM</td>
</tr>
<tr>
<td>ROLE</td>
<td>POSIT=551, OTHER=ANY, RACLST=DISALLOWED, MAXLNTH=246, GENLIST=DISALLOWED, DFTRETC=8, PROFDEF=YES, DFTUACC=NONE, OPER=NO, ID=1, FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>RRSFDATA</td>
<td>POSIT=65 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLREQUEST=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=NO DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO KEYQUAL=0 ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>RVARSMBR</td>
<td>POSIT=102 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLREQUEST=DISALLOWED MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=RACFVARS</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=73</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>SCDMBR</td>
<td>POSIT=9 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=39 DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=SECDATA</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=25</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>SCICSTST</td>
<td>POSIT=5 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=17 DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=UCICSTST</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=33</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY MAXLENX=25</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDSF</td>
<td>POSIT=100 OTHER=ANY RACL=ALLOWED MAXLNTH=63 GENLIST=ALLOWED DFTRETC=4 DFTUACC=NONE GROUP=GSDSF OPER=NO ID=94 FIRST=ANY</td>
</tr>
<tr>
<td>SECDATA</td>
<td>POSIT=9 OTHER=ALPHA MAXLNTH=8 DFTRETC=4 DFTUACC=NONE MEMBER=SCDMBR OPER=NO ID=26 FIRST=ALPHA</td>
</tr>
<tr>
<td>SECLABEL</td>
<td>POSIT=117 OTHER=ALPHANUM RACL=ALLOWED MAXLNTH=8 GENLIST=DISALLOWED DFTRETC=8 RACLREQ=YES DFTUACC=NONE MEMBER=SECLMBR OPER=NO ID=58 FIRST=ALPHA SIGNAL=YES GENERIC=DISALLOWED</td>
</tr>
<tr>
<td>SECLMBR</td>
<td>POSIT=117 OTHER=ANY RACL=DISALLOWED MAXLNTH=4 GENLIST=DISALLOWED DFTRETC=4 DFTUACC=NONE GROUP=SECLABEL OPER=NO ID=1 FIRST=ANY GENERIC=DISALLOWED</td>
</tr>
</tbody>
</table>
### CDT entries

#### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SERVAUTH</td>
<td>POSIT=558 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RAclist=ALLOWED  MAXlnth=64</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAclreq=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>EQUALMAC=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td></td>
<td>SIGNAL=YES</td>
</tr>
</tbody>
</table>

| SERVER | POSIT=546 OTHER=ANY |
|        | RAclist=ALLOWED  MAXlnth=41 |
|        | DFTRETC=8 |
|        | SLBLREQ=YES |
|        | EQUALMAC=YES |
|        | OPER=NO |
|        | ID=1 |
|        | FIRST=ALPHANUM |
|        | MAXLENX=64 |

| SFSCMD | POSIT=99 OTHER=ANY |
|        | RAclist=DISALLOWED  MAXlnth=246 |
|        | GENLIST=DISALLOWED DFTRETC=8 |
|        | DFTUACC=NONE |
|        | OPER=NO |
|        | ID=93 |
|        | FIRST=ANY |

| SIMS | POSIT=101 OTHER=ALPHANUM |
|      | RAclist=DISALLOWED  MAXlnth=8 |
|      | GENLIST=DISALLOWED DFTRETC=4 |
|      | DFTUACC=NONE |
|      | GROUP=UIMS |
|      | OPER=NO |
|      | ID=77 |
|      | FIRST=ALPHANUM |
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMESAGE</td>
<td>POSIT=116 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=0</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=59</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>SOMDOBS</td>
<td>POSIT=547 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>GROUP=GSOMDOB</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>STARTED</td>
<td>POSIT=66 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=17</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RA0LREQ=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>POSIT=122 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=50</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
</tbody>
</table>
## CDT entries

Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBSYSNM</td>
<td>( \text{POSIT}=83 ) ( \text{OTHER}=\text{ALPHANUM} )</td>
</tr>
<tr>
<td></td>
<td>( \text{RACLIST}=\text{ALLOWED} ) ( \text{MAXLNTH}=8 )</td>
</tr>
<tr>
<td></td>
<td>( \text{DFTRETC}=4 ) ( \text{DFTUACC}=\text{NONE} )</td>
</tr>
<tr>
<td></td>
<td>( \text{OPER}=\text{NO} ) ( \text{ID}=1 ) ( \text{FIRST}=\text{ALPHA} )</td>
</tr>
<tr>
<td>SURROGAT</td>
<td>( \text{POSIT}=104 ) ( \text{OTHER}=\text{ANY} )</td>
</tr>
<tr>
<td></td>
<td>( \text{RACLIST}=\text{ALLOWED} ) ( \text{MAXLNTH}=17 )</td>
</tr>
<tr>
<td></td>
<td>( \text{GENLIST}=\text{DISALLOWED} ) ( \text{DFTRETC}=4 )</td>
</tr>
<tr>
<td></td>
<td>( \text{DFTUACC}=\text{NONE} ) ( \text{OPER}=\text{NO} ) ( \text{ID}=71 ) ( \text{FIRST}=\text{ANY} )</td>
</tr>
<tr>
<td>SYSMVIEW</td>
<td>( \text{POSIT}=542 ) ( \text{OTHER}=\text{ANY} )</td>
</tr>
<tr>
<td></td>
<td>( \text{RACLIST}=\text{ALLOWED} ) ( \text{MAXLNTH}=52 )</td>
</tr>
<tr>
<td></td>
<td>( \text{GENLIST}=\text{DISALLOWED} ) ( \text{DFTRETC}=4 )</td>
</tr>
<tr>
<td></td>
<td>( \text{RACLRQ}=\text{YES} ) ( \text{DFTUACC}=\text{NONE} ) ( \text{OPER}=\text{NO} ) ( \text{ID}=1 ) ( \text{FIRST}=\text{ALPHANUM} )</td>
</tr>
<tr>
<td>SYSAUTO</td>
<td>( \text{POSIT}=598 ) ( \text{OTHER}=\text{ANY} )</td>
</tr>
<tr>
<td></td>
<td>( \text{RACLIST}=\text{ALLOWED} ) ( \text{MAXLNTH}=246 )</td>
</tr>
<tr>
<td></td>
<td>( \text{GENLIST}=\text{DISALLOWED} ) ( \text{DFTRETC}=4 )</td>
</tr>
<tr>
<td></td>
<td>( \text{RACLRQ}=\text{YES} ) ( \text{DFTUACC}=\text{NONE} ) ( \text{SLBLREQ}=\text{NO} ) ( \text{OPER}=\text{NO} ) ( \text{KEYQUAL}=0 ) ( \text{PROFDEF}=\text{YES} ) ( \text{ID}=1 ) ( \text{FIRST}=\text{ALPHANUM} ) ( \text{CASE}=\text{UPPER} ) ( \text{SIGNAL}=\text{NO} ) ( \text{GENERIC}=\text{ALLOWED} )</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAPEVOL</td>
<td>POSIT=1</td>
</tr>
<tr>
<td></td>
<td>OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=6</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=YES</td>
</tr>
<tr>
<td></td>
<td>ID=6</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>TCICSTRN</td>
<td>POSIT=5</td>
</tr>
<tr>
<td></td>
<td>OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=13</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>GROUP=GCICSTRN</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=12</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>TEMPDSN</td>
<td>POSIT=106</td>
</tr>
<tr>
<td></td>
<td>OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=DISALLOWED</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=NO</td>
</tr>
<tr>
<td></td>
<td>ID=69</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>TERMINAL</td>
<td>POSIT=2</td>
</tr>
<tr>
<td></td>
<td>OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLST=ALLOWED</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>GROUP=GTERMINL</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>EQUALMAC=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=7</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>SIGNAL=YES</td>
</tr>
<tr>
<td>Class</td>
<td>Attributes</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td>TIMS</td>
<td>POSIT=4, OTHER=ALPHANUM, MAXLTH=8, DFTRETC=4, DFTUACC=NONE, GROUP=GIMS, OPER=NO, ID=9, FIRST=ALPHANUM</td>
</tr>
<tr>
<td>TMEADMIN</td>
<td>POSIT=549, OTHER=ANY, RACLIST=ALLOWED, MAXLTH=246, GENLIST=ALLOWED, DFTRETC=8, DFTUACC=NONE, OPER=NO, ID=1, FIRST=ALPHANUM</td>
</tr>
<tr>
<td>TSOAUTH</td>
<td>POSIT=124, OTHER=ALPHANUM, RACLIST=ALLOWED, MAXLTH=8, GENLIST=DISALLOWED, DFTRETC=4, DFTUACC=NONE, OPER=NO, ID=48, FIRST=ALPHANUM</td>
</tr>
<tr>
<td>TSOPROC</td>
<td>POSIT=127, OTHER=ALPHANUM, RACLIST=ALLOWED, MAXLTH=8, GENLIST=DISALLOWED, DFTRETC=4, DFTUACC=NONE, OPER=NO, ID=45, FIRST=ALPHA</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCICSTST</td>
<td>POSIT=5 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=17</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=SCICSTST</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=34</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY MAXLENX=25</td>
</tr>
<tr>
<td>UIMS</td>
<td>POSIT=101 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=SIMS</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=78</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>UNIXMAP</td>
<td>POSIT=552 OTHER=NUMERIC</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>UNIXPRIV</td>
<td>POSIT=555 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>RACLREQ=YES SLBLREQ=NO</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>
### CDT entries

**Table 25. Classes supplied by IBM (continued)**

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCICSCMD</td>
<td>POSIT=5 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=21</td>
</tr>
<tr>
<td></td>
<td>DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=CCICSCMD</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=53</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>VMBATCH</td>
<td>POSIT=15 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>ID=24</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>VMBR</td>
<td>POSIT=120 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE GROUP=VMEVENT</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=54</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>VMCMD</td>
<td>POSIT=14 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=17</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>ID=22</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMDEV</td>
<td>POSIT=594, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED, MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAQREQ=NO, SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>RVRSMAC=NO, KEYQUAL=0</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=YES, ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>VMEVENT</td>
<td>POSIT=120, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED, MAXLNTH=16</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=VMBR</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=55</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
<tr>
<td>VMLAN</td>
<td>POSIT=64, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED, MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED, DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>VMMAC</td>
<td>POSIT=91, OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED, MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED, DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=91</td>
</tr>
<tr>
<td></td>
<td>PROFDEF=NO</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
</tbody>
</table>
## Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMMDISK</td>
<td>POSIT=18, OTHER=ANY, RACLIST=DISALLOWED, MAXLNTH=22, GENLIST=ALLOWED, DFTRETC=4, DFTUACC=NULL</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES, ID=20, FIRST=ANY</td>
</tr>
<tr>
<td>VMNODE</td>
<td>POSIT=16, OTHER=ANY, RACLIST=ALLOWED, MAXLNTH=8, GENLIST=ALLOWED, DFTRETC=4, DFTUACC=NULL</td>
</tr>
<tr>
<td></td>
<td>ID=23, FIRST=ANY</td>
</tr>
<tr>
<td>VMPOSIX</td>
<td>POSIT=63, OTHER=ANY, RACLIST=DISALLOWED, MAXLNTH=246, GENLIST=DISALLOWED, DFTRETC=4, DFTUACC=NULL</td>
</tr>
<tr>
<td></td>
<td>OPER=NO, ID=1, FIRST=ANY</td>
</tr>
<tr>
<td>VMRDR</td>
<td>POSIT=17, OTHER=ANY, RACLIST=DISALLOWED, MAXLNTH=17, GENLIST=ALLOWED, DFTRETC=4, DFTUACC=NULL</td>
</tr>
<tr>
<td></td>
<td>ID=21, FIRST=ANY</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMSEGMT</td>
<td>POSIT=90 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=92 FIRST=ANY</td>
</tr>
<tr>
<td>VMXEVENT</td>
<td>POSIT=96 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLST=DISALLOWED MAXLNTH=16</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=VXMBR OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=85 FIRST=ALPHA</td>
</tr>
<tr>
<td>VTAMAPPL</td>
<td>POSIT=114 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLST=ALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>RAQLREQ=YES DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=61 FIRST=ALPHANUM</td>
</tr>
<tr>
<td>VXMBR</td>
<td>POSIT=96 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLST=DISALLOWED MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE GROUP=VMXEVENT</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=84 FIRST=ALPHA</td>
</tr>
</tbody>
</table>
### CDT entries

**Table 25. Classes supplied by IBM (continued)**

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>WCICSRES</td>
<td>POSIT=5 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>MAXLNTH=40</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=RCICSRES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=53</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY CASE=ASIS</td>
</tr>
<tr>
<td>WIMS</td>
<td>POSIT=101 OTHER=ALPHANUM</td>
</tr>
<tr>
<td></td>
<td>RACLIST=DISALLOWED MAXLNTH=8</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=4</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>MEMBER=OIMS</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=82</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM</td>
</tr>
<tr>
<td>WRITER</td>
<td>POSIT=111 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=39</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=NONE</td>
</tr>
<tr>
<td></td>
<td>SLBLREQ=YES</td>
</tr>
<tr>
<td></td>
<td>RVRSMAC=YES</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=64</td>
</tr>
<tr>
<td></td>
<td>FIRST=ANY</td>
</tr>
<tr>
<td>XCSFKEY</td>
<td>POSIT=98 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=DISALLOWED DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>RACLI REQ=YES</td>
</tr>
<tr>
<td></td>
<td>GROUP=GXCSFKEY</td>
</tr>
<tr>
<td></td>
<td>OPER=NO</td>
</tr>
<tr>
<td></td>
<td>ID=100</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHA</td>
</tr>
</tbody>
</table>
### Table 25. Classes supplied by IBM (continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>XFACILIT</td>
<td>POSIT=8 OTHER=ANY</td>
</tr>
<tr>
<td></td>
<td>RACLIST=ALLOWED MAXLNTH=246</td>
</tr>
<tr>
<td></td>
<td>GENLIST=ALLOWED DFTRETC=8</td>
</tr>
<tr>
<td></td>
<td>DFTUACC=None</td>
</tr>
<tr>
<td></td>
<td>GROUP=GXFACILI</td>
</tr>
<tr>
<td></td>
<td>OPER=NO ID=1</td>
</tr>
<tr>
<td></td>
<td>FIRST=ALPHANUM SIGNAL=YES</td>
</tr>
</tbody>
</table>

| ZMFAPLA   | POSIT=592 OTHER=ANY            |
|           | RACLIST=ALLOWED MAXLNTH=246    |
|           | GENLIST=DISALLOWED DFTRETC=4   |
|           | RACLREQ=NO DFTUACC=None        |
|           | GROUP=GZMFAPLA SLBLREQ=NO      |
|           | OPER=NO ID=1 PROFDEF=YES       |
|           | FIRST=ANY CASE=ASIS           |
Appendix F. Requesting security services

The following is guidance information for application programmers requesting the services of the security product.

Application writers who need to use the services of a security product should try to write their programs in a way that is compatible with RACF. In turn, the developer of a security product other than RACF should provide function that is compatible with RACF. In this way, application programs should function with RACF or with some other external security product.

Because the functions of RACF are enhanced in new releases, an application might require a specific minimum level of RACF support in order to work correctly. If this is true, you should document this minimum level so that users of your application who have RACF installed know what level they need. Also, other security-product implementers then know what level of function they must provide to work correctly with your application. And users of other security products know what level they need to install to run your application.

Your application should use RACROUTE rather than the older macros RACHECK, RACDEF, RACLST, RACXTRT, RACINIT, FRACHECK, and RACSTAT.

Some Programming Interfaces to the security product might not be supported by all RACF-compatible security products. For example, the ICHEINTY interface (described in z/OS Security Server RACF Macros and Interfaces) is a General-Use interface that is only supported by RACF. If you want your application to work for users of RACF and of other security products, you should not make use of interfaces that are RACF-specific. The RACF-specific General-Use interfaces in this documentation are:

- The ACEEFCGP pointer to the list-of-groups table in the ACEE control block.
- The RACROUTE REQUEST=EXTRACT,TYPE=EXTRACT or TYPE=REPLACE functions that specify FIELDS= with a segment name (specified or implied) of BASE. You should not assume that any security product except RACF supports the extraction or replacement of named field information in the base segment of a profile.

The other security products, however, should support extraction or replacement of data in the other segments, such as DFP and TSO, if they are to be compatible with RACF. They should also support all the other functions of RACROUTE REQUEST=EXTRACT, for RACF compatibility.

As long as you avoid the RACF-specific interfaces, your program should work with any RACF-compatible security product. If you receive an SAF return code of 4 (in register 15) with a return code and reason code of 0 in SAFPRRET and SAFPRREA, that indicates that there is no security product, or the product does not support the request you are making. As an application designer, you are then free to make your own decision to continue with your processing, or to terminate processing.
Appendix G. Accessibility

Accessible publications for this product are offered through the z/OS Information Center, which is available at [www.ibm.com/systems/z/os/zos/bkserv/](http://www.ibm.com/systems/z/os/zos/bkserv/).

If you experience difficulty with the accessibility of any z/OS information, please send a detailed message to mhvrdfs@us.ibm.com or to the following mailing address:

IBM Corporation
Attention: MHVRCFS Reader Comments
Department H6MA, Building 707
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Accessibility features

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size.

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to [z/OS TSO/E Primer](http://www.ibm.com/systems/z/os/zos/bkserv/zos_tsoe_primer/) and [z/OS ISPF User's Guide Vol I](http://www.ibm.com/systems/z/os/zos/bkserv/zos_ispf_user_guide_vol_i/) for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users accessing the z/OS Information Center using a screen reader. In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), they can appear on the same line, because they can be considered as a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that your screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number 3.1) are mutually

© Copyright IBM Corp. 1994, 2013
exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, you know that your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The * symbol can be used next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *FILE with dotted decimal number 3 is given the format 3 *. FILE. Format 3* FILE indicates that syntax element FILE repeats. Format 3* * FILE indicates that syntax element * FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol giving information about the syntax elements. For example, the lines 5.1*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, this indicates a reference that is defined elsewhere. The string following the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you should refer to separate syntax fragment OP1.

The following words and symbols are used next to the dotted decimal numbers:

- ? means an optional syntax element. A dotted decimal number followed by the ? symbol indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that syntax elements NOTIFY and UPDATE are optional; that is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.

- ! means a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicates that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the same dotted decimal number can specify a ! symbol. For example, if you hear the lines 2! FILE, 2.1! (KEEP), and 2.1 (DELETE), you know that (KEEP) is the default option for the FILE keyword. In this example, if you include the FILE keyword but do not specify an option, default option KEEP will be applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, default FILE(KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1!
(KEEP), and 2.1.1 (DELETE), the default option KEEP only applies to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.

• * means a syntax element that can be repeated 0 or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1* data area, you know that you can include one data area, more than one data area, or no data area. If you hear the lines 3*, 3 HOST, and 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

Note:
1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you could write HOST STATE, but you could not write HOST HOST.
3. The * symbol is equivalent to a loop-back line in a railroad syntax diagram.

• + means a syntax element that must be included one or more times. A dotted decimal number followed by the + symbol indicates that this syntax element must be included one or more times; that is, it must be included at least once and can be repeated. For example, if you hear the line 6.1+ data area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. Similar to the * symbol, the + symbol can only repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loop-back line in a railroad syntax diagram.
Notices

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

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

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

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

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.
IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Site Counsel
IBM Corporation
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

COPYRIGHT LICENSE:

This information might contain sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Policy for unsupported hardware

Various z/OS elements, such as DFSMS, HCD, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted
for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

---

### Minimum supported hardware

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: [IBM Lifecycle Support for z/OS](http://www.ibm.com/software/support/systemsz/lifecycle/)
- For information about currently-supported IBM hardware, contact your IBM representative.

---

### Programming interface information

This documents intended Programming Interfaces that allow the customer to write programs to obtain the services of an external security manager.

This also documents information that is NOT intended to be used as Programming Interfaces of an external security manager. This information is identified where it occurs, either by an introductory statement to a chapter or section, or by the following marking:

```
NOT programming interface information
```

```
End of NOT programming interface information
```

---

### RSA Secure code

This product contains code licensed from RSA Data Security Incorporated.

---

### Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml)

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

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.
Index

A
abend
  JES abend 389
accessibility 459
contact IBM 459
features 459
ACEE data area
  changing 232, 315
  creating with RACROUTE REQUEST=VERIFY macro 206
administration, RACF
classroom courses xii
assistive technologies 459

C
CDT (class descriptor table)
  classes supplied by IBM 397
classroom courses, RACF xii
coding macros 4
combination field definitions
  definition 349
  format 349
connect template
  contents 365
continuation coding
  example 6
  continuation lines 6
courses about RACF xii

data set template
  contents 367
database profile
  storage requirements 350, 351

E
ENVIR=VERIFY parameter
  on RACROUTE macro 385
ENVR data structure 164
ENVR object storage area 164
example
  of continuation coding 6
  exit routine
SAF router 385, 386
exits 383
external security product 391
requirements 392
return and reason codes 393

F
fields
  character 349
  date 348
  integer 349
  time 349

first-party authorization check 30
FRCHECK macro 270
  chart of parameters by release 272
  description 270
  list form 273
  used to check a user's authorization to a resource 270
from RACROUTE REQUEST=STAT return codes 181

G
general resource
  fields in the profile 370
general template
  contents 370
group
  maximum number of users in 351
template for database 351
group profile
  description of the fields 352
group template
  contents 352

I
ICHERCDE macro
  class descriptor table supplied by IBM 397
ICHFR00 module
  receiving control from the SAF router 385
ICHFX02 exit routine
  reason codes 144, 145, 273
ICHRRCDX table 397
ICHRTX00 exit 385
ICHSAFP macro on MVS 387
installation exit
  ICHRRTX00 383
  IRRSXT00 389
irrcent, irrmulti, and irrsitec user IDs
  digital certificates 118
  IRRSXT00 exit 389

J
JES (job entry subsystem)
  handling of the return codes from SAF 389
  possible abend because of a RACF abend 389

K
keyboard
  navigation 459
PF keys 459
shortcut keys 459

M
macro forms
  what they mean 1
macros
  list of 7
RACDEF macro 275
RACHECK macro 295
RACINIT macro 309
RACLST macro 321
RACROUTE macro 8
RACROUTE REQUEST=AUDIT macro 23
RACROUTE REQUEST=AUTH macro 30
RACROUTE REQUEST=DEFINE macro 57
RACROUTE REQUEST=DIRAUTH 87
RACROUTE REQUEST=EXTRACT macro 98
RACROUTE REQUEST=FASTAUTH macro 135
RACROUTE REQUEST=LIST macro 148
RACROUTE REQUEST=SIGNON macro 161
RACROUTE REQUEST=STAT macro 178
RACROUTE REQUEST=TOKENBLD macro 187
RACROUTE REQUEST=TOKENMAP macro 197
RACROUTE REQUEST=TOKENXTR macro 201
RACROUTE REQUEST=VERIFY macro 206
RACROUTE REQUEST=VERIFYX macro 245
RACSTAT macro 327
RACXTRT macro 330
that invoke RACF 7
mapping macros
ICHSGX1P 169
maximum number of users in group 351
MVS router 385
installation exit 383
MVS router exit 383

N
navigation
  keyboard 459
Notices 463

P
parameter lists
  modifying with the SAF router exit routine 385
SAF router exit 386
password
how the SAF router exit routine can affect validation 389
postprocessing exit routine
FRACHECK macro
reason codes 273
RACROUTE REQUEST=FASTAUTH macro
reason codes 144, 145
profile
contents of a data set profile 367
contents of a general resource profile 370
contents of a group profile 352
contents of a user profile 354
in database 348
repeat groups 348
profile checking 51, 54, 307

R
RACDEF macro 275
chart of parameters by release 287
description 275
example 289, 290
execute form 292
list form 290
return codes 288
standard form 276
syntax 276, 292
used to define, modify, or delete resource profiles 275
RACF
system macros 7, 269
RACF database
connect template 365
general template 370
group template 352
reserved template 381
user template 354
RACF macros
coding 4
FRACHECK macro 270
list of 269
sample 4
that invoke RACF 269
RACF system macros
forms of the macro 1
RACHECK macro 295
chart of parameters by release 303
description 295
example 305, 306
execute form 307
list form 306
profile checking 307
return codes 303
standard form 295
syntax 295, 306, 307
used to provide authorization checking 295
RACINIT macro (continued)
standard form 310
syntax 310
used to provide user identification and verification 309
RACLST macro
chart of parameters by release 323
description 321
example 324
execute form 325
list form 325
return codes 323
standard form 321
syntax 321
used to build in-storage profiles 321
RACROUTE macro 8, 383
chart of parameters by RACROUTE keywords 9
color of RACROUTE keywords by parameters 9
description 8
examples 18
execute form 19
invoking the SAF router 385, 386
list form 18
modify form 20
return codes 17
standard form 13
syntax 13
used to invoke SAF 8
RACROUTE REQUEST=AUDIT macro
description 23
example 47, 48
execute form 51
list form 49
modify form 54
profile checking 51, 54
return codes 44
return codes (class descriptor table-default) 47
standard form 31
syntax 31, 49, 51, 54
used to provide authorization checking 30
RACROUTE REQUEST=DEFINE macro
description 57
example 78, 79
execute form 81
list form 79
modify form 85
return codes 74
standard form 58
syntax 58, 81, 85
used to define, modify, or delete resource profiles 57
RACROUTE REQUEST=DIRAUTH macro
description 87
RACROUTE REQUEST=DIRAUTH macro (continued)
example 94
execute form 96
list form 95
modify form 97
return codes 92
standard form 88
syntax 88, 95, 96, 97
RACROUTE REQUEST=EXTRACT macro
description 98
example 111, 112, 123, 124, 127
execute form 131
list form 129
modify form 133
return codes 119
standard form 100
syntax 100, 129, 131, 133
used to retrieve fields from user profile 98
RACROUTE REQUEST=FASTAUTH macro
description 135
example 157
execute form 159
list form 158
modify form 160
return codes 155
standard form 149
syntax 149
used to check a user’s authorization to a resource 135
RACROUTE REQUEST=LIST macro
description 148
example 157
execute form 159
list form 158
modify form 160
return codes 155
standard form 149
syntax 149
used to build in-storage profiles 148
RACROUTE REQUEST=SIGNON macro
description 161
example 172, 173
return codes 169
syntax
execute form 175
list form 173
modify form 176
standard form 162
RACROUTE REQUEST=STAT macro
description 178
example 182, 183, 184
execute form 185
list form 184
modify form 186
used to determine if RACF is active 178
RACROUTE REQUEST=TOKENBLD macro
description 187
example 192
execute form 194
list form 193
modify form 195
return codes 191
standard form 187
syntax 187
used to build a UTOKEN 187
RACROUTE REQUEST=TOKENMAP macro
  description 197
  example 199
  execute form 200
  list form 199
  modify form 201
  return codes 198
  standard form 197
  syntax 197, 199, 200, 201
  used to access token fields 197

RACROUTE REQUEST=TOKENXTR macro
  description 201
  example 204
  execute form 204
  list form 204
  modify form 205
  return codes 203
  standard form 202
  syntax 202, 204, 205
  used to extract a token 201

RACROUTE REQUEST=VERIFY macro
  description 206
  example 235, 236
  execute form 239
  list form 236
  modify form 242
  return codes 232
  standard form 207
  syntax 246, 246, 263, 266
  used to provide user identification and verification 206

RACROUTE REQUEST=VERIFYX macro
  description 245
  example 261
  execute form 263
  list form 261
  modify form 266
  return codes 258
  standard form 246
  syntax 246, 261, 263, 266
  used to build a UTOKEN 245

RACSTAT macro
  chart of parameters by release 328
  description 327
  example 328
  execute form 329
  list form 328
  used to determine if RACF is active 327

RACXTRT macro
  chart of parameters by release 340
  description 330
  execute form 344
  list form 342
  return codes 340
  standard form 331
  syntax 331, 342, 344
  used to retrieve fields from user profile 330
  reason codes
    from ICHRFX02 exit routine 144, 145, 273
  repeat groups
    in database 348

REQUEST=VERIFY parameter
  on RACROUTE macro 385
  resowner field, DFP 104
  return codes
    from FRACHECK macro 272
    from RACDEF macro 288
    from RACHECK macro 303
    from RACINIT macro 316
    from RACLIST macro 323
    from RACROUTE macro 17
    from RACROUTE REQUEST=AUDIT macro 26
    from RACROUTE REQUEST=AUTH macro 44
    from RACROUTE REQUEST=AUTH macro (class descriptor table-related) 47
    from RACROUTE REQUEST=DEFINE macro 73
    from RACROUTE REQUEST=DIRAUTH macro 92
    from RACROUTE REQUEST=EXTRACT macro 119
    from RACROUTE REQUEST=FASTAUTH macro 144
    from RACROUTE REQUEST=FASTAUTH macro (class descriptor table-related) 145
    from RACROUTE REQUEST=LIST macro 155
    from RACROUTE REQUEST=SIGNON macro 169
    from RACROUTE REQUEST=STAT macro 181
    from RACROUTE REQUEST=TOKENBLD macro 191
    from RACROUTE REQUEST=TOKENMAP macro 198
    from RACROUTE REQUEST=TOKENXTR macro 203
    from RACROUTE REQUEST=VERIFY macro 232
    from RACROUTE REQUEST=VERIFYX macro 258
    from RACXTRT macro 340
    from SAF router exit routine 387
  router exit 383
  router table
    description 385
    what it does 385

SAF router (continued)
  exit routine 385
  how entered 386
  simulating a call to RACF 388
  SAF router exit routine
    how it receives control 385
    parameter list 386
    requirements for 386
    return codes 387
  simulating a call to RACF 388
  uses for 385
  second-party authorization check 30
  security
    external product 391
    security environment, user 161
    security product router module 391
    security topics for RACF
      classroom courses  xii
      sending comments to IBM  xv
      shortcut keys 459
    SAF return codes 17
    SAF router overridden by UTOKEN 187, 245
    storage requirement
      database profiles 350, 351
    Summary of changes  xvi
    system authorization facility 383
    system authorization facility (SAF) 383
    system macros
      what the forms mean 1

T
  templates
    connect 365
    data set 367
    general 370
    group 352
    reserved 381
    user 354
  templates for database combination field definitions 349
  format of field definitions 347
  repeat groups 348
  third-party authorization check 30

U
  unit of work
    security information propagated from use session 206
  user identification and verification
    using RACROUTE REQUEST=VERIFY macro 206
  user interface
    ISPF 459
    TSO/E 459
  user profile
    description of the fields 354
  user security environment (ACEE) 161
  user template contents 354
  users
    maximum number in group 351
  UTOKEN
    building 187, 245

Index 469
W
warning mode  44
Product Number: 5650-ZOS

Printed in USA